EAST SEARCH HISTORY

HITS	QUERY	DATABASE						
5	(US-6546122-$ or US-6134340-$).did. or (EP-918300-$).did.	US-PGPUB; USPAT; EPO; JPO; DERWENT; IBM_TDB						
2	20020150283	US-PGPUB; USPAT; EPO; JPO; DERWENT; IBM_TDB						
0	wo-200111577-$.did	US-PGPUB; USPAT; EPO; JPO; DERWENT; IBM_TDB						
1	wo-200111577-$.did.	US-PGPUB; USPAT; EPO; JPO; DERWENT; IBM_TDB						
1	wo-200184494-$.did.	US-PGPUB; USPAT; EPO; JPO; DERWENT; IBM_TDB						
1	wo-200106445-$.did.	US-PGPUB; USPAT; EPO; JPO; DERWENT; IBM_TDB						
1	de-19811332-$.did.	US-PGPUB; USPAT; EPO; JPO; DERWENT; IBM_TDB						
0	ep-0918300-$.did.	US-PGPUB; USPAT; EPO; JPO; DERWENT; IBM_TDB						
0	ep-199918300-$.did.	US-PGPUB; USPAT; EPO; JPO; DERWENT; IBM_TDB						
2	ep-918300-$.did.	US-PGPUB; USPAT; EPO; JPO; DERWENT; IBM_TDB						
2	jp-11143833-$.did.	US-PGPUB; USPAT; EPO; JPO; DERWENT; IBM_TDB						
1	jp-06301768-$.did.	US-PGPUB; USPAT; EPO; JPO; DERWENT; IBM_TDB						
87	jain and fingerprint and core	US-PGPUB; USPAT; EPO; JPO; DERWENT; IBM_TDB						

6	jain.in. and fingerprint and core	US-PGPUB; USPAT; EPO; JPO; DERWENT; IBM_TDB						
6	(US-6487306-$ or US-6314197-$ or US-6289112-$ or US-6263091-$ or US-6185318-$ or US-6049621-$).did.	USPAT						
6	S16 and core	US-PGPUB; USPAT; EPO; JPO; DERWENT; IBM_TDB						
4	S16 and normal	US-PGPUB; USPAT; EPO; JPO; DERWENT; IBM_TDB						
0	S16 and perpen$6	US-PGPUB; USPAT; EPO; JPO; DERWENT; IBM_TDB						
0	se-199900553-$.did.	US-PGPUB; USPAT; EPO; JPO; DERWENT; IBM_TDB						
1	se-9900553-$.did.	US-PGPUB; USPAT; EPO; JPO; DERWENT; IBM_TDB						
0	weibe-linus.in.	US-PGPUB; USPAT; EPO; JPO; DERWENT; IBM_TDB						
24	wiebe-linus.in.	US-PGPUB; USPAT; EPO; JPO; DERWENT; IBM_TDB						
24	wiebe-linus.in.	US-PGPUB; USPAT; EPO; JPO; DERWENT; IBM_TDB						
0	verdicom.as. and russo.in.	US-PGPUB; USPAT; EPO; JPO; DERWENT; IBM_TDB						
14	veridicom.as. and russo.in.	US-PGPUB; USPAT; EPO; JPO; DERWENT; IBM_TDB						
2	6681034.pn.	US-PGPUB; USPAT; EPO; JPO; DERWENT; IBM_TDB						

2	6798334.pn.	US-PGPUB; USPAT; EPO; JPO; DERWENT; IBM_TDB						
44	hsu.in. and trw.as.	US-PGPUB; USPAT; EPO; JPO; DERWENT; IBM_TDB						
13	hsu.in. and trw.as. and fingerprint.ti.	US-PGPUB; USPAT; EPO; JPO; DERWENT; IBM_TDB						
1	jp-06301768-$.did.	US-PGPUB; USPAT; EPO; JPO; DERWENT; IBM_TDB						
0	santoshi-iwata.in. and fujitsu.as. and "fingerprint collation".ti.	US-PGPUB; USPAT; EPO; JPO; DERWENT; IBM_TDB						
0	santoshi-iwata.in. and fujitsu.as. and "fingerprint collation"	US-PGPUB; USPAT; EPO; JPO; DERWENT; IBM_TDB						
0	satoshi-iwata.in. and fujitsu.as. and "fingerprint collation".ti.	US-PGPUB; USPAT; EPO; JPO; DERWENT; IBM_TDB						
2	satoshi.in. and fujitsu.as. and "fingerprint collation".ti.	US-PGPUB; USPAT; EPO; JPO; DERWENT; IBM_TDB						
4174	satoshi.in. and fujitsu.as.	US-PGPUB; USPAT; EPO; JPO; DERWENT; IBM_TDB						
10	satoshi.in. and fujitsu.as. and fingerprint.ti.	US-PGPUB; USPAT; EPO; JPO; DERWENT; IBM_TDB						
33	satoshi.in. and fingerprint.ti.	US-PGPUB; USPAT; EPO; JPO; DERWENT; IBM_TDB						
7	iwata.in. and fingerprint.ti.	US-PGPUB; USPAT; EPO; JPO; DERWENT; IBM_TDB						
64	niizaki.in. and fingerprint.ti.	US-PGPUB; USPAT; EPO; JPO; DERWENT; IBM_TDB						
64	niizaki.in. and fingerprint.ti. and fujitsu.as.	US-PGPUB; USPAT; EPO; JPO; DERWENT; IBM_TDB						

20	yokoyama.in. and fingerprint.ti. and fujitsu.as.	US-PGPUB; USPAT; EPO; JPO; DERWENT; IBM_TDB						
0	yokoyama.in. and fingerprint.ti. and fujitsu.as. and normal	US-PGPUB; USPAT; EPO; JPO; DERWENT; IBM_TDB						
21616	fingerprint.ti. fujitsu.as. and normal	US-PGPUB; USPAT; EPO; JPO; DERWENT; IBM_TDB						
0	yokoyama.in. and fingerprint.ti. and fujitsu.as. and perpe$6	US-PGPUB; USPAT; EPO; JPO; DERWENT; IBM_TDB						
17	fingerprint.ti. and fujitsu.as. and (normal or perpendicular)	US-PGPUB; USPAT; EPO; JPO; DERWENT; IBM_TDB						
1	(core same ((normal or perpendicular) with ridge)) and ((382/115-127.ccls. or 300/383.ccls. or 300/384.ccls. or 340/5.52.ccls. or 340/5.53.ccls. or 340/5.52.ccls. or 340/5.81.ccls. or 340/5.82.ccls. or 340/5.83.ccls. or 340/5.84.ccls. or 396/15.ccls. or 902/3.ccls. or 283/68-69.ccls.)) and @ad<"20000531"	US-PGPUB; USPAT; EPO; JPO; DERWENT; IBM_TDB						

1	core same ((normal or perpendicular) with ridge) and ((382/115-127.ccls. or 300/383.ccls. or 300/384.ccls. or 340/5.52.ccls. or 340/5.53.ccls. or 340/5.52.ccls. or 340/5.81.ccls. or 340/5.82.ccls. or 340/5.83.ccls. or 340/5.84.ccls. or 396/15.ccls. or 902/3.ccls. or 283/68-69.ccls.)) and @ad<"20000531"	US-PGPUB; USPAT; EPO; JPO; DERWENT; IBM_TDB							
0	core same ((normal or perpendicular) same (ridge with direction)) and ((382/115-127.ccls. or 300/383.ccls. or 300/384.ccls. or 340/5.52.ccls. or 340/5.53.ccls. or 340/5.52.ccls. or 340/5.81.ccls. or 340/5.82.ccls. or 340/5.83.ccls. or 340/5.84.ccls. or 396/15.ccls. or 902/3.ccls. or 283/68-69.ccls.)) and @ad<"20000531"	US-PGPUB; USPAT; EPO; JPO; DERWENT; IBM_TDB							

9	core and ((normal or perpendicular) same (ridge with direction)) and ((382/115-127.ccls. or 300/383.ccls. or 300/384.ccls. or 340/5.52.ccls. or 340/5.53.ccls. or 340/5.52.ccls. or 340/5.81.ccls. or 340/5.82.ccls. or 340/5.83.ccls. or 340/5.84.ccls. or 396/15.ccls. or 902/3.ccls. or 283/68-69.ccls.)) and @ad<"20000531"	US-PGPUB; USPAT; EPO; JPO; DERWENT; IBM_TDB						
14	core and ((normal or perpendicular or orthogonal) same (ridge with direction)) and ((382/115-127.ccls. or 300/383.ccls. or 300/384.ccls. or 340/5.52.ccls. or 340/5.53.ccls. or 340/5.52.ccls. or 340/5.81.ccls. or 340/5.82.ccls. or 340/5.83.ccls. or 340/5.84.ccls. or 396/15.ccls. or 902/3.ccls. or 283/68-69.ccls.)) and @ad<"20000531"	US-PGPUB; USPAT; EPO; JPO; DERWENT; IBM_TDB						

8	core and ((orthogonal) same (ridge with direction)) and ((382/115-127.ccls. or 300/383.ccls. or 300/384.ccls. or 340/5.52.ccls. or 340/5.53.ccls. or 340/5.52.ccls. or 340/5.81.ccls. or 340/5.82.ccls. or 340/5.83.ccls. or 340/5.84.ccls. or 396/15.ccls. or 902/3.ccls. or 283/68-69.ccls.)) and @ad<"20000531"	US-PGPUB; USPAT; EPO; JPO; DERWENT; IBM_TDB						
12	core same curvature and ((382/115-127.ccls. or 300/383.ccls. or 300/384.ccls. or 340/5.52.ccls. or 340/5.53.ccls. or 340/5.52.ccls. or 340/5.81.ccls. or 340/5.82.ccls. or 340/5.83.ccls. or 340/5.84.ccls. or 396/15.ccls. or 902/3.ccls. or 283/68-69.ccls.)) and @ad<"20000531"	US-PGPUB; USPAT; EPO; JPO; DERWENT; IBM_TDB						

0	(singularit$6 same ((normal or perpendicular or orthogonal) with ridge)) and ((382/115-127.ccls. or 300/383.ccls. or 300/384.ccls. or 340/5.52.ccls. or 340/5.53.ccls. or 340/5.52.ccls. or 340/5.81.ccls. or 340/5.82.ccls. or 340/5.83.ccls. or 340/5.84.ccls. or 396/15.ccls. or 902/3.ccls. or 283/68-69.ccls.)) and @ad<"20000531"	US-PGPUB; USPAT; EPO; JPO; DERWENT; IBM_TDB						
3	singularit$6 same curvature and ((382/115-127.ccls. or 300/383.ccls. or 300/384.ccls. or 340/5.52.ccls. or 340/5.53.ccls. or 340/5.52.ccls. or 340/5.81.ccls. or 340/5.82.ccls. or 340/5.83.ccls. or 340/5.84.ccls. or 396/15.ccls. or 902/3.ccls. or 283/68-69.ccls.)) and @ad<"20000531"	US-PGPUB; USPAT; EPO; JPO; DERWENT; IBM_TDB						

2	singularit$6 and ((normal or perpendicular or orthogonal) same (ridge with direction)) and ((382/115-127.ccls. or 300/383.ccls. or 300/384.ccls. or 340/5.52.ccls. or 340/5.53.ccls. or 340/5.52.ccls. or 340/5.81.ccls. or 340/5.82.ccls. or 340/5.83.ccls. or 340/5.84.ccls. or 396/15.ccls. or 902/3.ccls. or 283/68-69.ccls.)) and @ad<"20000531"	US-PGPUB; USPAT; EPO; JPO; DERWENT; IBM_TDB						
0	(singularit$6 same ((normal or perpendicular or orthogonal))) and ((382/115-127.ccls. or 300/383.ccls. or 300/384.ccls. or 340/5.52.ccls. or 340/5.53.ccls. or 340/5.52.ccls. or 340/5.81.ccls. or 340/5.82.ccls. or 340/5.83.ccls. or 340/5.84.ccls. or 396/15.ccls. or 902/3.ccls. or 283/68-69.ccls.)) and @ad<"20000531"	US-PGPUB; USPAT; EPO; JPO; DERWENT; IBM_TDB						

9	singularit$6 and curvature and ((382/115-127.ccls. or 300/383.ccls. or 300/384.ccls. or 340/5.52.ccls. or 340/5.53.ccls. or 340/5.52.ccls. or 340/5.81.ccls. or 340/5.82.ccls. or 340/5.83.ccls. or 340/5.84.ccls. or 396/15.ccls. or 902/3.ccls. or 283/68-69.ccls.)) and @ad<"20000531"	US-PGPUB; USPAT; EPO; JPO; DERWENT; IBM_TDB						
8	US-4135147-$.DID. OR US-4947442-$.DID. OR US-4790564-$.DID. OR US-6763127-$.DID.	US-PGPUB; USPAT; EPO; JPO; DERWENT; IBM_TDB						
6	US-4185270-$.DID. OR US-4646352-$.DID. OR US-4944021-$.DID.	US-PGPUB; USPAT; EPO; JPO; DERWENT; IBM_TDB						
2	20020031245	US-PGPUB; USPAT; EPO; JPO; DERWENT; IBM_TDB						
2	20030039382	US-PGPUB; USPAT; EPO; JPO; DERWENT; IBM_TDB						
1	(US-6241288-$).did.	USPAT						
1	S63 and (core or singul$6)	US-PGPUB; USPAT; EPO; JPO; DERWENT; IBM_TDB						
3	Bergenek.in. and (core or singul$6)	US-PGPUB; USPAT; EPO; JPO; DERWENT; IBM_TDB						
1	(US-6241288-$).did.	USPAT						
1	S66 and (reference near2 point$3)	US-PGPUB; USPAT; EPO; JPO; DERWENT; IBM_TDB						
5	Bergenek.in. and (reference near2 point$3)	US-PGPUB; USPAT; EPO; JPO; DERWENT; IBM_TDB						
1	(US-6356649-$).did.	USPAT						

1	S69 and (core or singul$7)	US-PGPUB; USPAT; EPO; JPO; DERWENT; IBM_TDB						
2	5465303.pn.	US-PGPUB; USPAT; EPO; JPO; DERWENT; IBM_TDB						
0	5465303.pn. and (core or singul$6)	US-PGPUB; USPAT; EPO; JPO; DERWENT; IBM_TDB						
2	5140642.pn.	US-PGPUB; USPAT; EPO; JPO; DERWENT; IBM_TDB						
1	(US-6233348-$).did.	USPAT						
1	S74 and center	US-PGPUB; USPAT; EPO; JPO; DERWENT; IBM_TDB						
0	S74 and curv$6	US-PGPUB; USPAT; EPO; JPO; DERWENT; IBM_TDB						
1	20020150283 and normal	US-PGPUB; USPAT; EPO; JPO; DERWENT; IBM_TDB						
1	20020150283 and (predeterm$5 or predef$6) and normal	US-PGPUB; USPAT; EPO; JPO; DERWENT; IBM_TDB						
1	20020150283 and perip$8	US-PGPUB; USPAT; EPO; JPO; DERWENT; IBM_TDB						
1	20020150283 and (predeterm$5 or predef$6)	US-PGPUB; USPAT; EPO; JPO; DERWENT; IBM_TDB						
2	6719200.pn.	US-PGPUB; USPAT; EPO; JPO; DERWENT; IBM_TDB						
439357	6719200.pn. and smart card	US-PGPUB; USPAT; EPO; JPO; DERWENT; IBM_TDB						
2	6719200.pn. and "smart card"	US-PGPUB; USPAT; EPO; JPO; DERWENT; IBM_TDB						

1	6719200.pn. and thin$6	US-PGPUB; USPAT; EPO; JPO; DERWENT; IBM_TDB						
2	US-5862064-$.DID.	US-PGPUB; USPAT; EPO; JPO; DERWENT; IBM_TDB						
1	(US-5040224-$).did.	USPAT						
1	(US-5040224-$).did.	USPAT						
1	S87 AND (predetermined predefined threshold)	US-PGPUB; USPAT; EPO; JPO; DERWENT; IBM_TDB						
1	20020150283 and correl$6	US-PGPUB; USPAT; EPO; JPO; DERWENT; IBM_TDB						
23	(US-6241288-$ or US-6233348-$ or US-5040224-$ or US-6356649-$ or US-5915035-$ or US-6134340-$ or US-5140642-$ or US-5974163-$ or US-5465303-$ or US-6798334-$ or US-6681034-$ or US-6002784-$ or US-6181807-$ or US-5920641-$ or US-6719200-$).did. or (EP-918300-$).did. or (JP-06301768-$).did. or (JP-11143833-$ or DE-19811332-$ or WO-200106445-$ or WO-200184494-$ or US-20020030359-$).did.	USPAT; EPO; JPO; DERWENT						
8	S91 and correl$6	US-PGPUB; USPAT; EPO; JPO; DERWENT; IBM_TDB						
6	S91 and correl$6 and (align$6 or correct$6)	US-PGPUB; USPAT; EPO; JPO; DERWENT; IBM_TDB						
6	(US-6241288-$ or US-5040224-$ or US-6356649-$ or US-6134340-$ or US-5465303-$ or US-6181807-$).did.	USPAT						

	6	S94 and (align$6 or correct$6)	US-PGPUB; USPAT; EPO; JPO; DERWENT; IBM_TDB						
	4	S91 and (align$6)	US-PGPUB; USPAT; EPO; JPO; DERWENT; IBM_TDB						
	443	(registration or alignment) same (correlat$5) and fingerprint	US-PGPUB; USPAT; EPO; JPO; DERWENT; IBM_TDB						
	120	(registration or alignment) same (correlat$5) with (subregion or sub?region or region or subimage or sub?image or neighborhood or neighbourhood or window) and fingerprint	US-PGPUB; USPAT; EPO; JPO; DERWENT; IBM_TDB						
	35	(registration or alignment) same (correlat$5) with (subregion or sub?region or region or subimage or sub?image or neighborhood or neighbourhood or window) and fingerprint and "382".clas. and @ad<"20000531"	US-PGPUB; USPAT; EPO; JPO; DERWENT; IBM_TDB						
	35	(registration or alignment) same (correlat$5) with (subregion or sub?region or region or subimage or sub?image or neighborhood or neighbourhood or window) and S99	US-PGPUB; USPAT; EPO; JPO; DERWENT; IBM_TDB						

35	(US-5138468-$ or US-5067162-$ or US-6567566-$ or US-5040223-$ or US-6553129-$ or US-6813366-$ or US-6775392-$ or US-6700990-$ or US-6567533-$ or US-6560349-$ or US-6542618-$ or US-6539095-$ or US-6496591-$ or US-6449379-$ or US-6404898-$ or US-6400827-$ or US-6381341-$ or US-6363159-$ or US-6353672-$ or US-6343138-$ or US-6289108-$ or US-6266430-$ or US-6122403-$ or US-6122392-$ or US-6026193-$ or US-5862260-$).did. or (US-5850481-$ or US-5832119-$ or US-5822436-$ or US-5768426-$ or US-5748783-$ or US-5748763-$ or US-5745604-$ or US-5710834-$ or US-5636292-$).did.	USPAT						
35	(registration or alignment) same (correlat$5) with (subregion or sub?region or region or subimage or sub?image or neighborhood or neighbourhood or window) and S101	US-PGPUB; USPAT; EPO; JPO; DERWENT; IBM_TDB						
1	(US-5067162-$).did.	USPAT						
1	(US-5067162-$).did. and align and displacement	USPAT						
1	(US-5067162-$).did. and align$5 and displacement	USPAT						
1	(US-5067162-$).did. and align$5 and (displacement or transl$8)	USPAT						
1	(US-5067162-$).did. and (displacement or transl$8)	USPAT						

1	20020150283 and periph$6	US-PGPUB; USPAT; EPO; JPO; DERWENT; IBM_TDB						
2	5,959,541.pn.	US-PGPUB; USPAT; EPO; JPO; DERWENT; IBM_TDB						

EAST SEARCH TAGGED

patent #	type	pub date	title	class1	class2	inventor			
JP 11143833 A	DERWENT	19990528	User confirmation system used in entrance and exit of room - has communication unit which allows communication between sensor and IC card after receiving notice from comparator that data obtained by sensor corresponds to data held in data retainer						
US 6241288 B1	USPAT	20010605	Fingerprint identification/verification system	283/67	283/69; 283/78; 382/116; 382/124; 396/15	Bergenek; Jerker et al.			
US 5067162 A	USPAT	19911119	Method and apparatus for verifying identity using image correlation	382/126	382/209; 382/278	Driscoll, Jr.; Edward C. et al.			
US 6233348 B1	USPAT	20010515	Fingerprint registering apparatus, fingerprint identifying apparatus, and fingerprint identifying method	382/125		Fujii; Yusaku et al.			
US 5040224 A	USPAT	19910813	Fingerprint processing system capable of detecting a core of a fingerprint image by statistically processing parameters	382/124	382/228	Hara; Masanori			
US 6356649 B2	USPAT	20020312	Systems and methods with identity verification by streamlined comparison and interpretation of fingerprints and the like	382/115	356/71; 382/116; 382/191; 382/254; 382/275	Harkless; Curt R. et al.			
US 5915035 A	USPAT	19990622	Method for extracting high-level features for fingerprint recognition	382/125	382/205; 382/279	Hsiao; Pei-Yung et al.			
EP 918300 A2	EPO	19990526	Fingerprint feature correlator			HSU, SHI-PING et al.			

US 6134340 A	USPAT	20001017	Fingerprint feature correlator	382/124	382/125; 382/258	Hsu; Shi-Ping et al.			
US 5140642 A	USPAT	19920818	Method and device for allocating core points of finger prints	382/124	382/205	Hsu; Wen H. et al.			
JP 06301768 A	JPO	19941028	FINGERPRINT COLLATION DEVICE			IWATA, SATOSHI et al.			
US 5974163 A	USPAT	19991026	Fingerprint classification system	382/125	382/228	Kamei; Toshio			
US 5465303 A	USPAT	19951107	Automated fingerprint classification/identification system and method	382/124	382/125	Levison; Laurence L. et al.			
DE 19811332 A	DERWENT	19990923	Method of checking a biometric characteristic satisfies very high safety standard and can be implemented at supportable cost			MEISTER, G et al.			
US 6798334 B1	USPAT	20040928	Method and device for verifying a biometric characteristic	340/5.52	340/5.53	Meister; Gisela et al.			
WO 200106445 A	DERWENT	20010125	Finger print templates matching method for identifying individuals, involves loading one reference data chunk and one measurement data chunk in random access memory for comparison			RUSSO, A P			
US 6681034 B1	USPAT	20040120	Method and system for fingerprint template matching	382/125	902/6	Russo; Anthony P.			
US 6002784 A	USPAT	19991214	Apparatus and method for detecting features of a fingerprint based on a set of inner products corresponding to a directional distribution of ridges	382/124		Sato; Atsushi			

US 6181807 B1	USPAT	20010130	Methods and related apparatus for fingerprint indexing and searching	382/124	707/6; 707/7	Setlak; Dale R. et al.			
US 5920641 A	USPAT	19990706	Method for reconstructing linear structures present in raster form	382/125	382/115; 382/118; 382/119; 382/124; 382/126; 382/128; 382/173; 382/177; 382/179	Ueberreiter; Birgit et al.			
WO 200184494 A	DERWENT	20011108	Biometric identity check using a portable data carrier with a biometric template for comparing to a biometric sample collected by a biometric sensor			WALFRIDSSON, K et al.			
US 20020030359 A	DERWENT	20020314	Access right checking system for electronic money transaction through Internet, has smart card whose processor compares preprocessed biometric data with reference data to determine if accessing right exists			WIEBE, L et al.			
US 6719200 B1	USPAT	20040413	Checking of right to access	235/382	235/382.5; 340/5.1; 356/71; 382/115; 382/124	Wiebe; Linus			

Miscellaneous Search and Discovery

US006356649B2

(12) **United States Patent**
Harkless et al.

(10) **Patent No.: US 6,356,649 B2**
(45) **Date of Patent: *Mar. 12, 2002**

(54) **"SYSTEMS AND METHODS WITH IDENTITY VERIFICATION BY STREAMLINED COMPARISON AND INTERPRETATION OF FINGERPRINTS AND THE LIKE"**

(75) Inventors: **Curt R. Harkless**, Canoga Park; **Randall E. Potter**, North Hills; **John A. Monro, Jr.**, Alhambra; **Lawrence R. Thebaud**, Agoura Hills, all of CA (US)

(73) Assignee: **Arete Associate, Inc.**, Sherman Oaks, CA (US)

(*) Notice: This patent issued on a continued prosecution application filed under 37 CFR 1.53(d), and is subject to the twenty year patent term provisions of 35 U.S.C. 154(a)(2).

Subject to any disclaimer, the term of this patent is extended or adjusted under 35 U.S.C. 154(b) by 0 days.

(21) Appl. No.: **08/843,219**

(22) Filed: **Apr. 11, 1997**

(51) **Int. Cl.**^{7} **G06K 9/00**

(52) **U.S. Cl.** **382/115**; 382/116; 382/191; 382/254; 382/275; 356/71

(58) **Field of Search** 382/115, 116, 382/124, 125, 190, 191, 201, 206, 209, 254, 260, 275; 356/71; 340/825.34; 250/208.1, 216

(56) **References Cited**

U.S. PATENT DOCUMENTS

3,112,468 A	*	11/1963	Kamentsky	382/125
5,859,420 A	*	1/1999	Borza	250/208.1
5,909,501 A	*	6/1999	Thebaud	382/124
5,963,657 A	*	10/1999	Bowker et al.	382/127

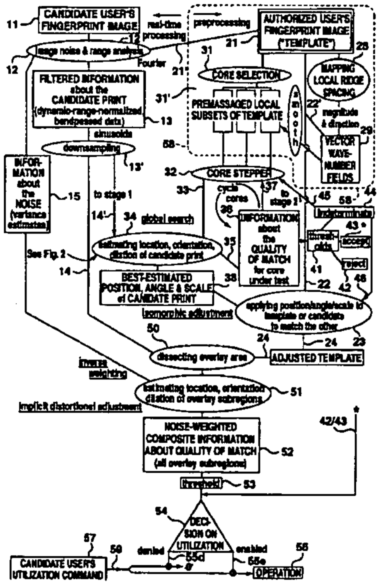
* cited by examiner

Primary Examiner—Matthew C. Bella
Assistant Examiner—Sheela Chawan
(74) *Attorney, Agent, or Firm*—Smart & Biggar

(57) **ABSTRACT**

Preferably a sensor receives a print image from an authorized person to form a template, and from a candidate to form test data. Power spectral density (PSD) data for the template and candidate are compared, to read out rotation & dilation; these are used to adjust the template or candidate preparatory to a correlation to find translation. After applying the translation, and refinement of the rotation and dilation, normalized spatial correlation values (NSCVs) are used as a measure of quality of the match—and thresholded to make an early rejection or acceptance decision in very clear cases. Where the question is closer, isomorphic adjustment is applied to the entire template or candidate for a fairer comparison in their overlap area. Such comparison proceeds by the same type of PSD analysis—but for multiple subregions in the overlap area. Resulting NSCVs are averaged to obtain a measure of quality of the match, which again is thresholded for a final decision in the closer cases. Noise variance from the test data, vs. position in the image, is used to weight the importance of comparison with the template in each subregion. Nonvolatile memory holds instructions for automatic operation.

22 Claims, 8 Drawing Sheets


11
CANDIDATE USER'S FINGERPRINT IMAGE
real-time processing
preprocessing
AUTHORIZED USER'S FINGERPRINT IMAGE ("TEMPLATE")
12
21
28
image noise & range analysis
12
Fourier
21'
31
CORE SELECTION
MAPPING LOCAL RIDGE SPACING
FILTERED INFORMATION about the CANDIDATE PRINT (dynamic-range-normalized bandpassed data)
13
31'
PREMASSAGED LOCAL SUBSETS OF TEMPLATE
smooth
22'
magnitude & direction
29
sinusoids
downsampling
58
VECTOR WAVE-NUMBER FIELDS
13'
INFORMATION about the NOISE (variance estimates)
15
32
CORE STEPPER
33
cycle cores
37
to stage 2
45
58
44
to stage 1
14'
34
global search
36
INFORMATION about the QUALITY OF MATCH for core under test
indeterminate
43
thresholds
accept
estimating location, orientation, dilation of candidate print
35
See Fig. 2
reject
14
41
BEST-ESTIMATED POSITION, ANGLE & SCALE of CANIDATE PRINT
38
22
42
46
applying position/angle/scale to template or candidate to match the other
isomorphic adjustment
50
24
24
23
dissecting overlay area
ADJUSTED TEMPLATE
inverse weighting
estimating location, orientation dilation of overlay subregions
51
implicit distortional adjustment
42/43
NOISE-WEIGHTED COMPOSITE INFORMATION ABOUT QUALITY OF MATCH (all overlay subregions)
52
threshold
53
54
DECI-SION ON UTILIZATION
57
59
denied
enabled
56
55d
55e
CANDIDATE USER'S UTILIZATION COMMAND
OPERATION

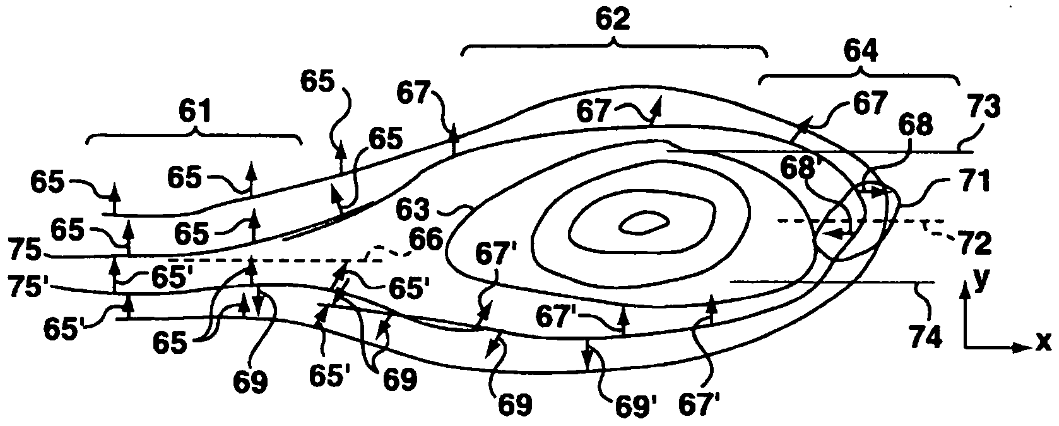


FIG. 7

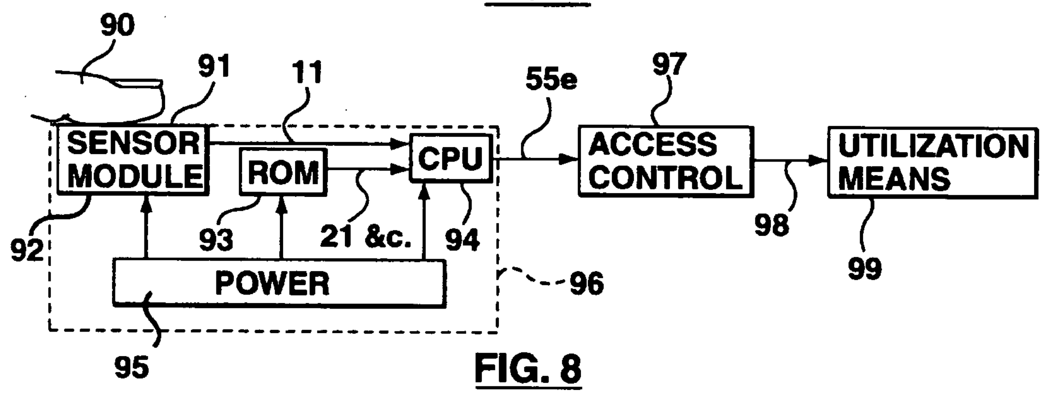


FIG. 8

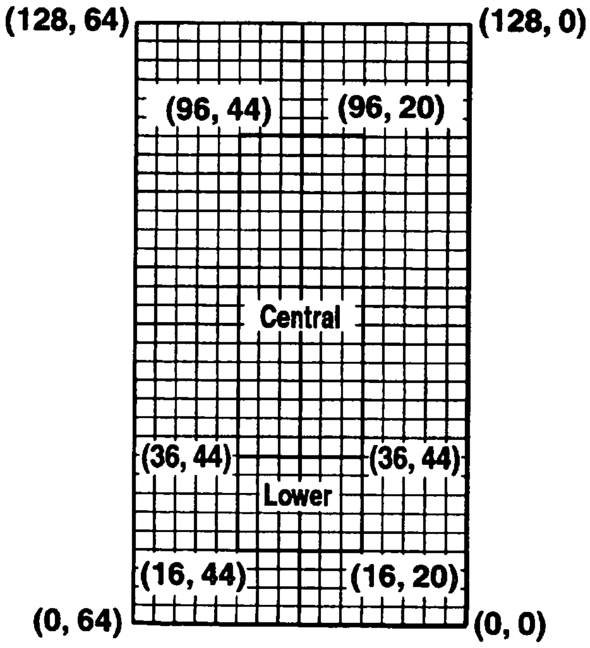


FIG. 9

US 20030039382A1

(19) **United States**

(12) **Patent Application Publication** (10) **Pub. No.: US 2003/0039382 A1**

Yau et al. (43) **Pub. Date: Feb. 27, 2003**

(54) **FINGERPRINT RECOGNITION SYSTEM**

(75) Inventors: **Shing-Tung Yau**, Belmont, MA (US); **Xianfeng Gu**, Cambridge, MA (US); **Zhiwu Zhang**, Malden, MA (US)

Correspondence Address:
WEINGARTEN, SCHURGIN, GAGNEBIN &
LEBOVICI LLP
TEN POST OFFICE SQUARE
BOSTON, MA 02109 (US)

(73) Assignee: **Biometric Informatics Technolgy, Inc.**

(21) Appl. No.: **10/156,447**

(22) Filed: **May 28, 2002**

Related U.S. Application Data

(60) Provisional application No. 60/293,487, filed on May 25, 2001. Provisional application No. 60/338,949, filed on Oct. 22, 2001.

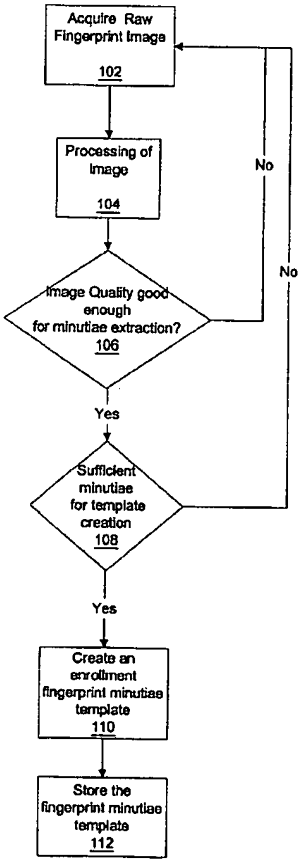
Publication Classification

(51) **Int. Cl.**^{7} **G06K 9/00**

(52) **U.S. Cl.** **382/125**

(57) **ABSTRACT**

A method of analyzing and recognizing fingerprint images that utilizes vector processing of a vector field that is defined as the tangential vector of the fingerprint ridge curves is disclosed. The raw fingerprint image is divided into blocks, filtered to remove noise, and the orientation direction of each block is found. This allows the ridge curves to be enhanced and approximated by piece-wise linear approximations. The piece-wise linear approximations to the ridge curves allow the minutiae to be extracted and classified and a fingerprint minutiae template to be constructed. An enrollment process gathers multiple fingerprint images, creates fingerprint minutiae templates corresponding to the acquired fingerprint images, and stores the templates and other data associated with the respective individual or the enrolled fingerprint in a fingerprint database. In an identification process, an unknown raw fingerprint image is obtained via a fingerprint scanner and processed similarly to the enrollment process such that the fingerprint minutiae template of the unknown fingerprint is compared to one or more previously enrolled fingerprint minutiae templates. The identity of the individual associated with the unknown fingerprint is thereby ascertained. In addition, live finger detection can be accomplished in conjunction with the verification or identification process through analysis of the fingerprint image thus enhancing the security of the overall system.


Acquire Raw Fingerprint Image
102
Processing of Image
104
No
No
Image Quality good enough for minutiae extraction?
106
Yes
Sufficient minutiae for template creation
108
Yes
Create an enrollment fingerprint minutiae template
110
Store the fingerprint minutiae template
112

ACTAtek

HECTRIX LTD.
SMARTCARD, BIOMETRIC & INTERNET APPLIANCE TECHNOLOGIES
Email: tech@hectrix.com , info@hectrix.com
Website: www.hectrix.com ; www.actatek.com ; www.ibonus.com

FINGERPRINT NOTES

Technical Information

Features	Technical Specification
Image Resolution:	500DPI
False Rejection Rate (FRR):	1%
False Acceptance Rate (FAR):	0.0001%
Allowable Fingerprint Rotation:	+/-15degree
Operation Temperature:	-25 to +65 Degrees Celsius
Number of minutiae being taken:	30 to 60 depending on user
Matching Speed:	0.05 second
Scanning Speed:	1.50 second

Introduction

ACTAtek ™ uses latest Optical Scanning technology with its own algorithms and matching calculations, a step above other sensors in the market.

It must be emphasized that to get an accurate enrollment and quick authentication each time a fingerprint is presented, the fingerprint placement must be towards the center of the scanner. Placing your finger far from the center position of the sensor will increase the rejection rate.

Finger Rotation should be kept to a minimum during enrollment and verification.

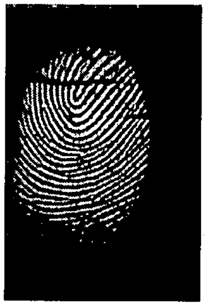
When enrolling, place the finger on the sensor where the entire core can clearly be seen by the scanner.

A **good** image is critical for the overall performance of the fingerprint scanner. Any deviation from a good image, either by placing the finger far away from the scanner, or by applying too much pressure or not locating it in the **CENTER** of the scanner, will cause the scanner's rejection rate to rise. Read below on how to get a good image for your enrollment/authentication.

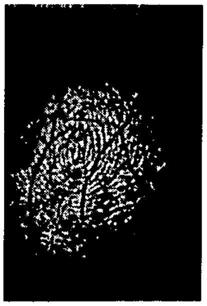
Good vs Bad Image

A good fingerprint image is one in which the core of the fingerprint is well-defined and easily recognizable. The core of a finger is defined as the "point located within the inner most recurving ridge". It is normally located in the MIDDLE of the fingerprint. It is therefore critical when enrolling that you place the finger on the scanner where the entire core can clearly be seen.

An example of a good & bad image is displayed as follows:



Good Image: A reading where the whole fingerprint core can be seen clearly.

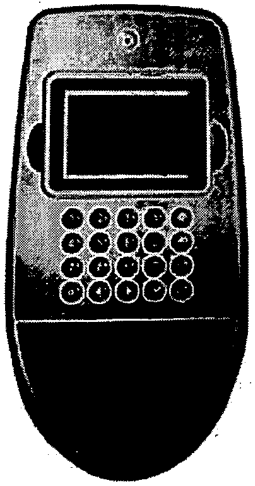


Bad Image: An image where the crackles & displacement of the fingerprint core decries it unrecognizable.

FINGERPRINT NOTES

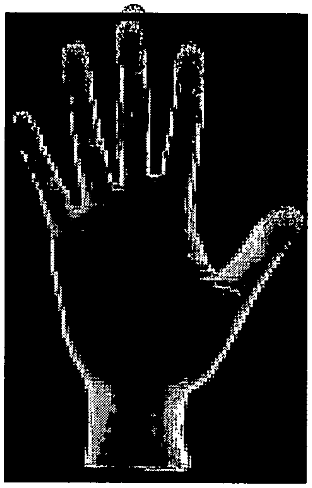
Fingerprint Enrollment & Authentication

In order to receive a successful enrollment and authentication, it is critical that the following be noted carefully. Each successful enrollment will result in a successful authentication and save a lot of time in troubleshooting and erroneous readings.



Optical Fingerprint Scanner

Fingerprint Core.



It is highly recommended for the fingerprint core be big and clear for a successful enrollment of a clear and good image.

Make sure the fingerprint image captured is of the core of the finger presented. A fingerprint core is a point located within the innermost recurving ridge of any given finger.

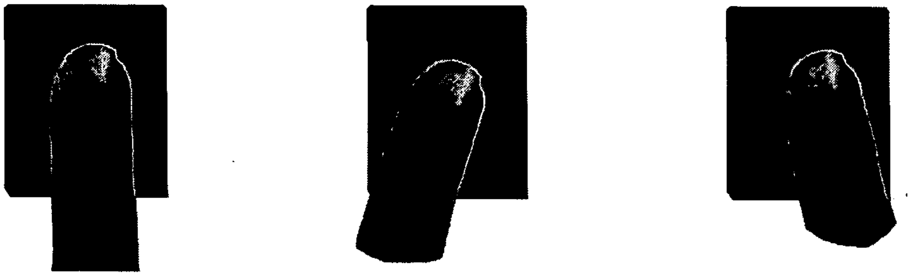
Also, to obtain a higher success rate, enroll the same finger 3 times in a slightly adjusted angle, one to the center, one inclined slightly to the left and the third inclined slightly to the right.

If you follow the following enrollment procedure, the success rate will increase dramatically.

FINGERPRINT NOTES

Fingerprint Enrollment:

Step 1: Place the center of any one finger directly above the sensor in the dead center, as shown below:



Step 2: Place the center of the same finger (enrolled in Step 1), slightly aligned to the left.

Step 3: Place the center of the same finger, slightly aligned to the right.

After each placement, wait for the message "Template Stored" on the LCD screen to appear, and then remove your finger and press "Enter/Return" to enroll the second or third finger(s).

If you have any questions regarding the enrollment procedure, e-mail us at tech@hectrix.com .

Singular Point Detection in Fingerprint Images Using Linear Phase Portraits

Sharat Chikkerur[1], Sharath Pankanti[2],Nalini Ratha[2] and Venu Govindaraju[1]

[1]Center for Unified Biometrics and Sensors, University at Buffalo, NY, USA
[2]IBM T. J. Watson Research Center, Hawthorne, NY, USA
{ssc5,govind}@buffalo.edu,{sharat,ratha}@us.ibm.com

Abstract. A fingerprint image represents a system of oriented texture, marked by a directed flow of ridges and valleys. The qualitative nature of this oriented flow is determined by the position and type of singular points such as the core and delta. The detection of the singular points and estimation of its position forms an important step in fingerprint image analysis and has several useful applications such as classification, registration and orientation flow modeling. We present a novel approach for estimation of singular point position using linear phase portraits. Linear phase portraits are based on the geometric theory of differential equations and provide a qualitative description of vector flows. The proposed method has several advantages over existing schemes based on Poincaré index including robustness to noise. We present an objective evaluation of the algorithm using images from FVC2002 DB1 database.

1 Introduction

2005

The fingerprint image represents a system of a strongly oriented texture consisting of a directed system of ridges and valleys. Minutiae features such as ridge ending and bifurcation mark local discontinuities in the ridge flow and constitute the local features. These features have been used extensively in fingerprint verification [5]. On the other hand, the orientation image and the singular points form the global features that give information about the coarser trends within the image. The orientation image captures the dominant direction of the ridge flow in each region of the fingerprint image. The singular points like 'core' and 'delta' mark the areas of interest within this flow. Figure 1 shows the local and global features of a typical fingerprint image. Among these features, the extraction of the singular points forms an important step in fingerprint image processing and has several useful applications. The classification of the ridge flow ('arch','tented arch','whorl' or 'loop') can be directly determined based on the number, type and positions of the singular points. The singular points also represent the intrinsic points of reference within the fingerprint image and can therefore be used to register or approximately align two different images during matching. Finally, there exist linear and non-linear methods [8,11,13] to accurately estimate the orientation of the fingerprint image given complete information about the singular points. However, estimating the location of the singular points itself is a difficult problem. Existing methods are based on computing the Poincaré index of the orientation image at each pixel position [1,4]. These methods are sensitive to noise and generate many false positives in noisy images. In this paper, we present a robust algorithm based on linear phase portraits. Linear phase portraits are based on the geometric theory of differential equations and provide a qualitative description of vector flows and have traditionally used to qualitatively analyze vector fields arising from fluid flows. The technique cannot be directly applied to images due to inherent ambiguity in the direction of the edges. However, subsequent work by Rao and Jain[6,7], Shu et al[9], led to the development of several techniques that allow their use for analysis of oriented textures. In this paper, we further adapt the process for analysis of fingerprint images.

The rest of the paper is organized as follows. Section 1.1 discusses the prior done in this area. Section 2 presents an overview of the proposed approach. The concept of linear phase portraits and its application to fingerprint images is discussed in detail in section 3. The results of the experiments are in Section 4. And finally, the conclusions are discussed in Section 5.

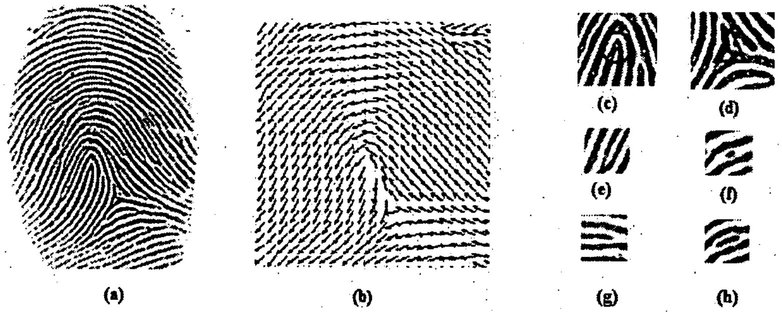


Figure 1 Illustration of fingerprint features. Global features: (b) Orientation image , singular points Core(c) and Delta (d). Local minutiae features: Bifurcation(e), Ridge dot(f), Ridge ending (g) and Ridge island(h).

1.1 Prior Related Work

There have been several approaches for the detection of singular points in literature. By far, the most popular method is the one proposed by Kawagoe and Tojo [4] and is based on the computation of the Poincaré index. The Poincaré index for any given point in a vector field is computed by summing the orientation difference between successive elements along a closed path around that point. It can be shown that around closed curves the Poincaré index assumes only one of the discrete values 0°,±180°,±360°. The singular regions can be easily detected since the singularities corresponding to the loop, whorl and delta assume a Poincaré index value of 180°,360° and -180° respectively . Recently, an interesting improvisation on the Poincaire index was proposed by Bazen and Gerez [1]. In this method, the line integration around each point is avoided by using the result of Green's theorem which states that, a line integral around any point in a vector field can be replaced by a surface integral of the curl of the vector field [12]. However, methods based on Poincaire index are very sensitive to noise and lead to detection of false singularities in low-quality images.

Srinivasan and Murthy [10] proposed another approach based on the examination of the local characteristics of the orientation image. They examine the dominant direction of the orientation in four quadrants around each point and use several rules to detect and classify the singular points. This method is very heuristic in its approach. A more elegant approach was suggested by Capelli et al. [2]. They proposed a irregularity operator that indicates the local deviation of the orientation around each point. The operator is applied on each pixel in the image to yield an irregularity map. Singularities can be extracted by examining the dark regions within the map. However, the extraction requires considerable processing of the irregularity map. They also proposed a novel partitioning based approach, where the ridge flow is segmented into regions of consistent orientation by an iterative clustering algorithm. Singular points are determined by considering the locations where the region borders intersect.

2 Overview

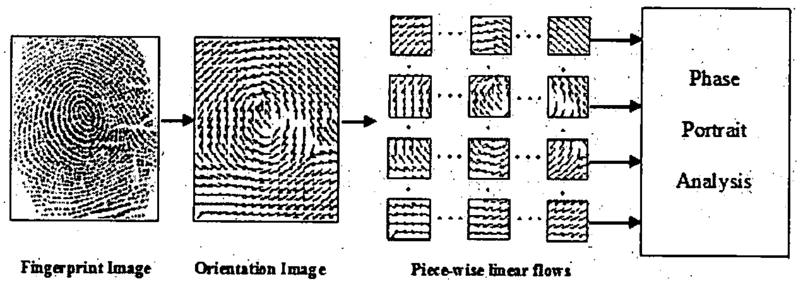


Figure 2 Overview of the proposed approach

Figure 2 shows the overview of the proposed approach. The orientation image describing the ridge flow is obtained using well known methods outlined in [6]. While it is difficult to model the entire orientation image through any linear methods, it can be modeled as a combination of piece-wise linear flows. Each piece wise linear flow is obtained by moving a sliding window of size (WxW) across the orientation image. In our case W=7. At each position of the window, we derive a parametric representation of the local flow using linear phase portraits. This representation provides us not only with the possible location of the singularity, but also about the nature of the singularity. More details are presented in section 3. As each piece-wise linear flow is analyzed, evidence is accumulated about the possible location and type of the singularities in a spirit similar to generalized Hough transformation [3]. The two dimensional accumulator array can be treated as an intensity image, where each position indicates the likelihood of being a singular point. This image is processed using grey-scale morphology and connected component analysis to obtain a best estimate of the type and position of each of the singular points. The proposed approach of analyzing the orientation locally has several advantages over global fingerprint models such as

1. The analysis assumes the orientation image to be piecewise linear, and can therefore handle any arbitrary flow pattern
2. The parameter estimation is done using a least square minimization approach rendering the process very robust to noise
3. The analysis provides an over-complete description that can be used to reconstruct a smooth estimate of the ridge flow.

3 Singular Point Detection

In this section, we present an outline of linear phase portraits that is sufficient for analysis of fingerprint images and also explain its application to singular point detection. A more complete discussion of linear phase portraits is presented in [6,7,9].

3.1 Linear Phase Portraits

Phase portraits are based on the geometric theory of differential equations and are used to derive qualitative description of vector flows and fluid flows in particular. Consider a system of linear differential equations in two dimensions described by ,

$$\begin{bmatrix} \dot{x} \\ \dot{y} \end{bmatrix} \doteq \begin{bmatrix} \left(\frac{dx}{dt}\right) \\ \left(\frac{dy}{dt}\right) \end{bmatrix} = \begin{bmatrix} a_{11} & a_{12} \\ a_{21} & a_{22} \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix} + \begin{bmatrix} b_1 \\ b_2 \end{bmatrix} \quad (1)$$

In general, for N dimensions, $[\dot{X}] = AX + B, X = [x_1, x_2..x_N]^T$ (2)

The two functions x(t) and y(t) represent a curve in the (x,y) plane. The differential equation specified in Eqn. (1) specifies how the curve evolves with t. The geometrical representation of the qualitative behavior of the curve is known as the *phase portrait.* It can be shown that for a linear system of equations only a finite number of qualitative phase portraits are possible [6], with the other instances being related by a equivalence relation. The nature of the flow can be determined through its local properties defined by

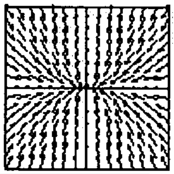
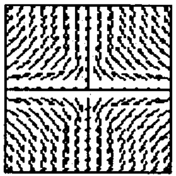
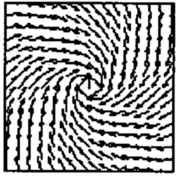
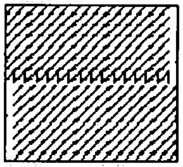
$$curl(\dot{X}) = (a_{21} - a_{12}) \quad (3)$$

$$div(\dot{X}) = trace(A) = (a_{11} + a_{22}) \tag{4}$$

$$def(\dot{X}) = \sqrt{(a_{11} - a_{22})^2 + (a_{21} + a_{12})^2} \tag{5}$$

$$\Delta = \sqrt{def(\dot{X})^2 - curl(\dot{X})^2} \tag{6}$$

Table 1. Shows the possible configuration of phase portraits and their required condition. Of all the possible configuration, only the spiral and the saddle type of phase portraits are relevant to fingerprint images since they correspond directly with loop/whorl and delta type of singularities

Flow Type	Example	Condition
Node		$\lvert div(\dot{X}) \rvert > \Delta\ and\ \ \Delta > 0$
Saddle		$\lvert div(\dot{X}) \rvert < \Delta\ and\ \ \Delta > 0$
Spiral		$\lvert div(\dot{X}) \rvert \neq 0\ and\ \ \Delta^2 < 0$
Parallel		|A|=0 (A is singular)

3.2 Parameter Estimation

The orientation image is divided into a system of piecewise linear flows, using a sliding window centered at each pixel in the orientation image. At each position the corresponding matrix A and B are determined. These represent the parameters of the system of linear differential equations that is capable of reproducing the local orientation. These parameters can be used to determine the location of the singular point as outlined in the next section.

In order to derive a parameters A,B for each local flow, we adopt a least square minimization approach that results in a synthesized flow with minimum deviation from the observed flow. In particular we try to minimize the distance measures specified by

$$D = \frac{\sum_{(i,j)\in W} \left|\sin(\theta - \hat{\theta})\right|}{2} \tag{7}$$

$$\hat{\theta}(x,y) = \arctan\left(\frac{\dot{y}}{\dot{x}}\right) = \arctan\left(\frac{a_{21}x + a_{22}y + b_2}{a_{11}x + a_{12}y + b_1}\right) \tag{8}$$

Here $\hat{\theta}(x,y)$ represents orientation from the synthesized flow and $\theta(x,y)$ represents the orientation in the observed flow. The distance metric D is invariant to ±180° ambiguity between $\hat{\theta}(x,y)$ and $\theta(x,y)$. Other distance metrics such as the acute angle between the lines with slopes $\hat{\theta}(x,y)$ and $\theta(x,y)$ given by equation (9) are also invariant to this phase ambiguity and may be used without affecting the nature of the solution. The parameters (a11,a12,a21,a22,b1,b2) are evaluated using Levenberg-Marquardt non-linear least squares minimization approach. More details of the implementation is provided in [6].

$$D = \sum_{(x,y)} \left| \frac{\tan\theta(x,y) - \tan\hat{\theta}(x,y)}{1 + \tan\theta(x,y)\tan\hat{\theta}(x,y)} \right| \tag{9}$$

The least squares optimization approach renders the estimation fairly robust to noise, a common source of errors in the Poincaré index based approaches. The resilience of the methods is shown in **Figure 3** where the flow is reconstructed in the presence of Gaussian noise in the orientation image.

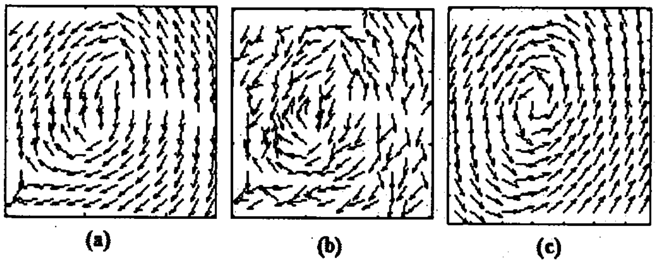


Figure 3 (a)Original orientation flow (b) Flow in presence of noise (c) Reconstructed flow using linear phase portrait analysis.

3.3 Position Estimation

Given the linear phase portrait description, the singularities represent critical points where the curve [x(t),y(t)] is stationary. This is specified by the condition,

$$\left[\dot{X}\right] = AX + B = 0 \qquad (10)$$

The position of the singular point is given by,

$$X_c = -\left(A^{-1}B\right) \qquad (11)$$

However, all types of singularities satisfy this condition. In order to estimate the positions of the core and delta positions separately we make use of the classification provided in Table 1. As we analyze each section of the orientation image, an estimate of the possible location for each singularity is obtained based on Eqn.(9). We use this information to update an accumulator that represents the likelihood of each cell being a singularity. Separate accumulators are maintained for core(spiral) and delta(saddle) singularities. This is similar to the voting approach used in Generalized Hough Transform. However, unlike Hough Transform, we also increment the vote of the neighboring cells in a weighted fashion to make the estimate more robust to noise.

3.4 Singularity Map Processing

The resulting position maps I(x,y) can be treated as grey scale images with each pixel intensity proportional to the likelihood of the presence of a singularity. In order to estimate the position of the singular points the following algorithm is used individually on both the spiral and the saddle map resulting from the phase portrait analysis.

1. The intensity image is binarized using both a relative and an absolute threshold T_1 and T_2. Here T_1 represents threshold on the relative likelihood of the pixel being a singular point and T2 represents the absolute threshold indicating the minimum vote required by a pixel for it to be considered as a candidate position. The abso-

lute threshold is required to prevent the detection of false singular points in instances where they are not present in the image at all. This case arises when the fingerprint is placed on the sensor such that the core or delta(s) does not appear in the image. However, due to noise, some linear neighborhoods may be incorrectly classified as candidates for core or delta and may lead to a false detection. The absolute threshold helps us to eliminate such spurious candidates. The binary image M(x,y) is obtained using the rule,

$$M(x,y) = \begin{cases} 1 \text{ if } I(x,y) > \max(T_1, T_2) \\ \quad 0 \text{ otherwise} \end{cases} \tag{12}$$

$$T_1 = \max\{I(x,y)\},\ T_2 = 10$$

2. Possible locations where the singularity may lie is obtained by performing connected component analysis of the thresholded image. Each component is positioned roughly centered on each singularity. Outliers that have too few components or too many components are rejected. The resulting map consists of candidate locations each associated with
3. The positions of the singular points are determined as the centroids of the candidate locations. For each connected component C

4 Experimental Evaluation

We evaluated the algorithm using 100 random images from FVC2002 DB1 database. The images were cropped to remove the background prior to the processing. Fig. 3 and **Figure 4** show the results over two such images.

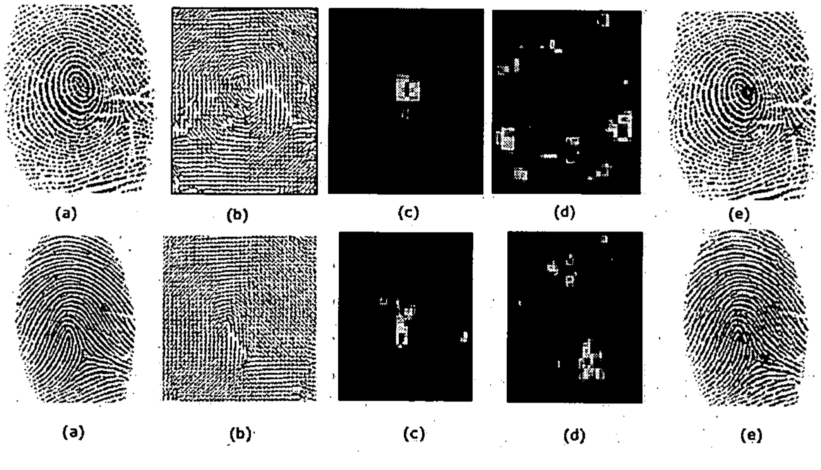


Figure 4 (a)Original Image (b) Orientation Image (c) Spiral map (d) Saddle map (e) Results of the spiral/saddle map processing.

In order to evaluate the accuracy of the proposed method, we estimated the position of the singular points in each of the 100 images and compared it with the ground truth locations manually marked by the authors. The deviation is measured in terms of the absolute distance between the estimated and actual location of the singularities and is shown for each sample. Also, the algorithm fails to detect singular points in certain cases. A majority of these instances are where the singular points are located at the edge of the fingerprint foreground and the absence of the surrounding regions make it difficult to detect the deviation in the flow. The summary results are shown in **Table 2**. It can be seen that on an average the core and delta position can be localized within a ridge width.

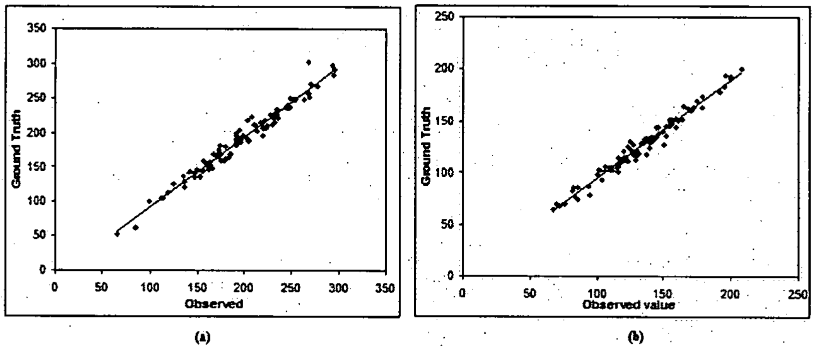


Figure 5 (a) Regression line for X position of the core location (b) Regression line for Y position of the core position. Similar plot can be obtained for the delta location and have been omitted for brevity.

Table 2 Summary results of the algorithm

Number of core points missed by the algorithm	8 of 102
Number of spurious core points	1 of 102
Number of delta points missed by the algorithm	5 of 52
Number of spurious delta points	5 of 52
Mean distance from the actual core (dx,dy) (pixels)	(9.10,7.24)
Mean distance from the actual delta (dx,dy) (pixels)	(5.72,11.32)

5 Conclusion and Future Work

In this paper we presented a novel approach for estimation of singular point position in fingerprint images using linear phase portraits. The proposed approach presents several advantages over existing methods based on Poincare index and in particular its robustness to noise and its ability to provide a qualitative description of the flow at each point. We also presented an objective evaluation of the algorithm over 100 images of FVC2002 DB1 database and showed that it provides a very accurate estimate of the singular point positions. Future work will also involve providing an optimal reconstruction algorithm for the orientation image based on the piece-wise linear analysis. Currently, the flow parameters evaluated at each location in the orientation image leading to an over determined system of piece-wise linear flows. We will explore the optimal sampling strategy in order to reduce computational complexity.

References

1. Bazen A. M. and Gerez S. H., "Systematic Methods for the Computation of Directional Fields and Singular Points of Fingerprints", IEEE Transactions on PAMI, Vol. 24, No. 7, July 2002
2. Cappelli R., Lumini A., Maio D., Maltoni D.,"Fingerprint Classification by Directional Image Partitioning", IEEE PAMI, Vol. 21 No.5, pp.402-421, 1999
3. Duda R. O, Hart P. E. "Use of the Hough transformation to detect lines and curves in pictures", Communications of the ACM, Vol. 15, No. 1, 1972
4. Kawagoe M., Tojo A., "Fingerprint Pattern Classification", Pattern Recognition, Vol. 17 No. 3, pp. 295-303, 1984.
5. Maltoni D., Maio D., Jain A. K. and Prabhakar S, "Handbook of Fingerprint Verification", Springer Verlag, 2003
6. Rao R., "A Taxonomy for Texture Description and Identification", Springer-Verlag 1990.
7. Rao R., and Jain R., "Analyzing Oriented Textures through Phase Portraits" ICPR 1990, pp. 336-340.
8. Sherlock B. and Monro D. "A Model for Interpreting Fingerprint Topology", *Pattern Recognition*, Vol. 26, No. 7, pp. 1047-1055, 1993.
9. Shu C. F., Jain R., and Quek F., "A linear Algorithm for Computing the Phase Portraits of Oriented Textures," , CVPR 1991, pp. 352-357.
10. Srinivasan V. S., Murthy N. N., "Detection of Singular Points in Fingerprint Images", Pattern Recognition, Vol. 25 No.2, pp.139-153, Feb. 1992.
11. Vizcaya P. and Gerhardt L. A Nonlinear Orientation Model for Global Description of Fingerprints", Pattern Recognition, Vol. 29, No. 7, pp. 1221-1231, 1996.
12. Weisstein E. W, "Green's Theorem", http://mathworld.wolfram.com
13. Zhou J, Gu J., Modeling Orientation Fields of Fingerprints with Rational Complex Functions", Pattern Recognition Vol. 37, No. 2, pp. 389-391, 2004.

Curvature Estimation in Oriented Patterns Using Curvilinear Models Applied to Gradient Vector Fields

Joost van de Weijer, Lucas J. van Vliet,
Piet W. Verbeek, and
Michael van Ginkel, *Member, IEEE*

Abstract—Curved oriented patterns are dominated by high frequencies and exhibit zero gradients on ridges and valleys. Existing curvature estimators fail here. The characterization of curved oriented patterns based on translation invariance lacks an estimation of local curvature and yields a biased curvature-dependent confidence measure. Using parameterized curvilinear models we measure the amount of local gradient energy along the model gradient as a function of model curvature. Minimizing the residual energy yields a closed-form solution for the local curvature estimate and the corresponding confidence measure. We show that simple curvilinear models are applicable in the analysis of a wide variety of curved oriented patterns.

Index Terms—Oriented patterns, anisotropy, curvature, confidence measures, curvilinear models, gradient vector fields.

1 Introduction

Reliable estimation of local features in digitized images is of great importance for many image processing tasks (segmentation, analysis, and classification). Depending on the class of images under investigation, knowledge of different features is desired. One such class of images is defined by Kass and Witkin [1] as oriented patterns: patterns that exhibit a dominant local orientation. Examples are seismic, acoustic, wood grain, interference patterns, and fingerprint images. Important features for these images are estimates of local anisotropy, orientation, curvature, and scale.

The structure tensor yields a robust estimator for local orientation [1], [2], [3], [4] based on a local gradient vector field. This estimator locally models the images as translation invariant strokes. In addition to orientation estimation, this method also yields an anisotropy measure indicating the resemblance of the local area to a translation invariant model. This measure can also be interpreted as a confidence measure of the estimated orientation. Both a lack of smoothness (e.g., caused by noise) and deviations from the translation invariant model (e.g., curved oriented patterns) are responsible for a decrease of this confidence measure. To distinguish between the two possible causes, we proposed a parabolic transformation which optimizes the translation invariance after transformation [5]. This method yields a curvature estimate for curved oriented patterns as a by-product. A shortcoming of this method is that the proposed transformation is not orthonormal and, therefore, it lacks conservation of gradient energy. This does not allow direct comparison of the confidence values of different transformations. In this paper, we propose a method to investigate the resemblance of a local pattern of 2D oriented pattern to a certain model function (e.g., circular, parabolic). The model is represented by a parameterized transformation function of the isophotes. The method assures the conservation of gradient energy, allowing us to compare confidence measures of different transformations and, especially, of a parameterized transformation for different parameter values. As in [5], solving the parameter for best confidence yields a closed-form estimate of the additional free parameter, e.g., local curvature. We propose two curvilinear models, a parabolic and a circular model, for the characterization of curved oriented patterns. When the resemblance between a model and a local image is high, the corresponding model parameters, orientation, and curvature yield a reliable description of the local image. The method yields features with a corresponding confidence value. All these estimates are local and can be represented as feature maps.

Estimation of the curvature in oriented patterns is not trivial. Worring and Smeulders [6] presented an extensive comparison between curvature estimators applied to segmented data for which the position and ordering of points along the contour have to be known. For noisy oriented patterns, segmentation fails, making these methods useless. The isophote (tangential) curvature (the second derivative along the isophote divided by the gradient magnitude) and the normal curvature [17] are segmentation-free [7], [8], [17], but also fail on these images. There are three reasons for this [5]: 1) The gradient is zero on ridges and in valleys. 2) Increasing the regularization scale of directional derivatives suppresses the oriented pattern and reduces the signal-to-noise ratio. 3) Opposite sides of a ridge (or valley) yield curvatures of opposite sign which cancel out after averaging. The only two methods which do yield a curvature estimate for oriented patterns are either very computationally demanding [9] or are not accompanied by a confidence measure, which makes them hard to rely on [10].

normal
curvature
$\kappa \sim \vec{n}$

The proposed method resembles a method for the detection of complex symmetries as presented by Bigün et al. [11], [12], [13]. They characterize symmetries by (coordinate) transformation functions which transform symmetric patterns into translation invariant patterns. The success of such a transformation is determined by the confidence measure of the structure tensor applied to the transformed image. A high confidence value is an indicator for the presence of the corresponding symmetry. Bigün et al. method is an extension of the generalized Hough transform. Detection of a symmetry pattern involves accumulation of evidence by voting. Bigün's symmetry detector requires two orthonormal transformation functions. It measures the resemblance of the local differential field to two perpendicular differential fields (indicating the symmetry), whereas our method looks at the resemblance of the local differential field to only one differential field (representing the shape of the isophotes). This difference allows us to estimate model parameters by optimizing the resemblance between the actual differential field and a model differential field in a closed-form solution, i.e., omitting a time, consuming voting scheme. This is not possible with the symmetry method since neither one of the two differential fields is preferred.

- *J. van de Weijer is with the ISIS group of the Faculty of Science, University of Amsterdam, Kruislaan 403, 1098 SJ Amsterdam, The Netherlands. E-mail: joostw@wins.uva.nl.*
- *L.J. van Vliet, P.W. Verbeek, and M. van Ginkel are with the Pattern Recognition Group of the Faculty of Applied Sciences, Delft University of Technology, Lorentzweg 1, 2628 CJ Delft, The Netherlands. E-mail: {L.J.vanVliet, P.W.Verbeek, michael}@ph.tn.tudelft.nl.*

Manuscript received 22 July 1999; revised 2 Aug. 2000; accepted 17 Jan. 2001. Recommended for acceptance by S. Sarkar.
For information on obtaining reprints of this article, please send e-mail to: tpami@computer.org, and reference IEEECS Log Number 110288.

The requirement for two orthonormal transformation functions poses an unnecessary limitation to the symmetries. For example, such a set of functions does not exist for the parabolic model we propose, i.e., parabolic isophotes along a linearly increasing symmetry axis. We extend his method by noting that only the existence of the differential fields of the two transformation functions is essential.

2 Oriented Patterns

An oriented pattern $m(x,y)$ can be written as a real one dimensional function g of a model function u

$$m(x,y) = g(u(x,y,\mathrm{a})). \tag{1}$$

The model function $u(x,y,\mathrm{a})$ describes the shape of the isophotes and a contains local isophote parameters such as orientation and curvature. Consequently, the gradient (differential field) of m,

$$\nabla m = \frac{dg}{du}\nabla u, \tag{2}$$

is a dg/du weighted version of the gradient of u. In oriented patterns, we distinguish between two perpendicular orientations; along the isophote (tangent) and along the gradient. Note that orientation is defined on the interval $[0,\pi)$. Consequently, vectors in opposite directions have the same orientation.

Consider the function $f(x,y)$ representing a local image (window) and a model function $u(x,y,\mathrm{a})$. It is of interest to what extent $f(x,y)$ is described by the model function $u(x,y,\mathrm{a})$. This is measured by decomposing the derivative energy of $f(x,y)$ into two contributions, one parallel and one perpendicular to the normalized differential field of $u(x,y,\mathrm{a})$. This results in the following energies:

$$\begin{aligned} E_f(\mathrm{a}) &= \int\!\!\int \left(\nabla f \cdot \frac{\nabla u(\mathrm{a})}{\|\nabla u(\mathrm{a})\|}\right)^2 dx\,dy, \\ E_r(\mathrm{a}) &= \int\!\!\int \left(\nabla f \cdot \frac{(\nabla u(\mathrm{a}))_\perp}{\|\nabla u(\mathrm{a})\|}\right)^2 dx\,dy, \end{aligned} \tag{3}$$

where $E_f(\mathrm{a})$ denotes the fit energy and $E_r(\mathrm{a})$ the residual energy. The subscript $\perp$ indicates a rotation of 90° of the vector and the integrals represent the averaging over the local image. A quality measure of the fit can be found by comparing the fit energy with the residual energy. Since no a priori knowledge exists to interpret the energy difference between the fit and the residual energy, we normalize the difference with the total gradient energy to obtain the following quality measure $c(\mathrm{a})$:

$$c(\mathrm{a}) = \frac{E_f(\mathrm{a}) - E_r(\mathrm{a})}{E_f(\mathrm{a}) + E_r(\mathrm{a})} \qquad -1 \le c \le 1. \tag{4}$$

The value of $c(\mathrm{a})$ varies from -1 for a pattern of which the isophotes are exactly perpendicular to those of the model function $u(x,y,\mathrm{a})$ and +1 for a pattern which is exactly described by the model function. The isotropic noise energy is distributed equally between the fit and the residual energy.

More important than the quality measure for an arbitrary a is to know which a maximizes the quality function c, i.e., maximizes E_f and minimizes E_r. The vector a contains model parameters which describe local features. Therefore, optimizing the confidence function c corresponds to feature estimation. Furthermore, the quality measure $c(\mathrm{a})$ informs us about the success of the fit and can be seen as a confidence measure of the estimated features. Besides comparing confidence measures of the same model function, it is also possible to compare confidence measures of different model functions. Note that the normalization of the confidence measures is independent of the model function. By comparing optimized confidence functions of various models, one can find out which model describes the local pattern best.

Usually, the complexity of the confidence function does not allow a closed-form solution of the optimization criterion. The straight model is an exception. In the case of curvilinear models, we avoid costly (iterative) optimization procedures by considering approximate confidence functions which do allow closed-form solutions.

3 Straight-Oriented Patterns

Locally, many oriented patterns can be characterised by a straight model. For such a pattern the model function $u(x,y,\mathrm{a})$ is given by

$$u(x,y,\phi) = x\cos\phi + y\sin\phi, \tag{5}$$

with ϕ the orientation perpendicular to the **model** isophotes. Substituting this in (3) yields

$$E_f(\phi) = \frac{1}{2}\left(\overline{f_x^2} + \overline{f_y^2}\right) + \frac{1}{2}\left(\overline{f_x^2} - \overline{f_y^2}\right)\cos 2\phi + \frac{1}{2}\ \overline{2f_xf_y}\sin 2\phi. \tag{6}$$

A bar ($\overline{\bullet}$) denotes an averaged quantity and will from now on replace the integrals responsible for averaging over a local image. The confidence value $c(\phi)$ is

$$c(\phi) = \frac{1}{\overline{f_x^2} + \overline{f_y^2}}\left(\left(\overline{f_x^2} - \overline{f_y^2}\right)\cos 2\phi + \overline{2f_xf_y}\sin 2\phi\right).. \tag{7}$$

$c(\phi)$ can be maximized as a function of the orientation ϕ. This yields the following (gradient-based) orientation estimator [1], [2], [3], [4]:

$$\phi_{opt} = \frac{1}{2}\arctan\frac{2\overline{f_xf_y}}{\overline{f_x^2} - \overline{f_y^2}}. \tag{8}$$

with confidence value $c(\phi_{opt})$:

$$c(\phi_{opt}) = \frac{d^2}{g^2}, \qquad \text{where } d^4 = \overline{f_x^2 - f_y^2}^2 + \overline{2f_xf_y}^2. \tag{9}$$

This confidence measure can also be interpreted as a measure for translation invariance and shows an intuitive dependency to the pattern orientation ϕ_{opt}.

$$\begin{aligned} c(\phi) &= \frac{d^2(\cos^2(\phi - \phi_{opt}) - \sin^2(\phi - \phi_{opt}))}{g^2} \\ &= \frac{1}{2}\ c(\phi_{opt})(1 + \cos(2(\phi - \phi_{opt}))). \end{aligned} \tag{10}$$

The maximum of the confidence measure $c(\phi_{opt})$ reduces due to noise in the local image f. For a linear pattern p distorted by additive uncorrelated noise n $(f = p + n)$, the confidence value $c(\phi_{opt})$ is:

$$c(\phi_{opt}) = \frac{d^2}{\overline{\|\nabla f\|^2}} = \frac{d^2}{\overline{\|\nabla p + \nabla n\|^2}} = \frac{d^2}{\overline{\|\nabla p\|^2}}\,\frac{\overline{\|\nabla p\|^2}}{\overline{\|\nabla p\|^2} + \overline{\|\nabla n\|^2}}. \tag{11}$$

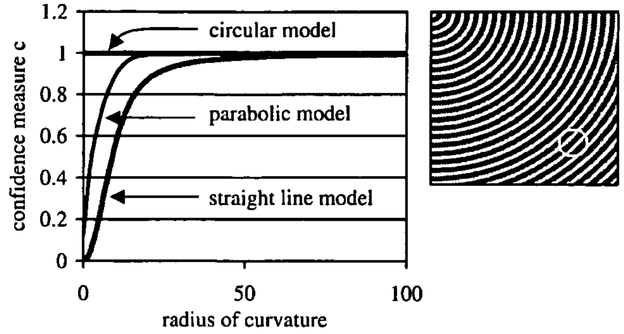


Fig. 1. Confidence measure $c(\hat{\phi}, \hat{\kappa})$ of circular, parabolic, and straight line models on a noise-free pattern of concentric circles.

Note that the gradient noise energy is divided equally over E_f and E_r. Therefore, the numerator of c is unaffected by noise. Noise increases the total gradient energy (denominator of c), which lowers the confidence value $c(\phi_{opt})$. Another reason for a lower confidence value is a deviation between the local image and the model function. For instance when curved lines occur, then curvature will contribute to E_r. In the next section, we will extend the model to include curved patterns.

4 Curved Oriented Patterns

We present two model functions which locally model curved oriented patterns. A parabolic model

$$u(x, y, \phi, \kappa) = \frac{1}{2}\ \kappa w^2 - v \qquad (12)$$

and a concentric circle model

$$u(x, y, \phi, \kappa) = \sqrt{\kappa^2 w^2 + (1 - \kappa v)^2} \qquad (13)$$

in which κ is the curvature. The Gauge coordinates v, w are obtained by

$$v = x \cos\phi + y \sin\phi \qquad w = -x \sin\phi + y \cos\phi \qquad (14)$$

Here, we discuss the parabolic approximation. For the circular approximation, we refer to Appendix A. Using the parábolic model function and (3), the following energies are obtained:

$$\begin{aligned} E_f(\phi, \kappa) &= \overline{\left(\frac{\kappa^2 w^2 f_w^2 - 2\kappa w f_v f_w + f_v^2}{1 + \kappa^2 w^2}\right)}, \\ E_r(\phi, \kappa) &= \overline{\left(\frac{\kappa^2 w^2 f_v^2 + 2\kappa w f_v f_w + f_w^2}{1 + \kappa^2 w^2}\right)}, \end{aligned} \qquad (15)$$

where f_v and f_w are the derivatives in, respectively, the v and w direction. Finding the curvature and orientation that maximize the confidence function requires a search in ϕ, κ-space. In this paper, we shall not further investigate this method due to its high computational demands. Instead, we propose a way to approximate the confidence function, allowing a fast closed-form solution.

An approximation to the orientation ϕ can be obtained by looking at the axis of minimal translation invariance for parabolic and circular patterns. In the case of a circular pattern, this is the v-axis. For a parabolic pattern, it depends on the curvature and the window size used. For small curvatures (i.e., compared to the window size), the minimal translation invariant axis is equal to the w-axis. Increasing the curvature, the axis of minimal translation

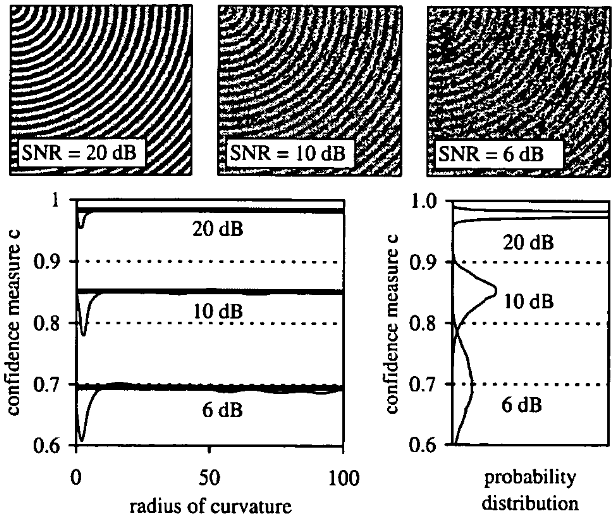


Fig. 2. (a) Average confidence measure $c(\hat{\phi}, \hat{\kappa})$ for the circular model as a function of the radius for three different SNRs (top to bottom: 20 dB, 10 dB, 6 dB). The measure $c(\hat{\phi}, \hat{\kappa})$ yields a small bias for small radii. The horizontal lines indicate the average confidence measure for, straight model for the corresponding SNR. (b) Probability density functions of the confidence measures for the straight-oriented patterns for the three different SNRs (top to bottom: 20 dB, 10 dB, 6 dB).

invariance jumps to the w-axis. Therefore, an approximation of the orientation needed to determine the v and w-axes in (15) can be computed with (8). After substituting the orientation, the resulting equations only depend on the curvature. Iterative maximization of the confidence function in κ-space is still time-consuming. We propose approximating this maximum by using locally adapted weighting. The weighting function of E_f and E_r (denoted by the bar $\bar{\bullet}$) is in its turn weighted by $(1 + \kappa^2 w^2)$ after which we normalize for this weighting. This mathematical trick has a high resemblance to normalized convolution [14]. It results in

$$\begin{aligned} \hat{E}_f(\kappa) &= \frac{\kappa^2 \overline{w^2 f_w^2} - 2\kappa \overline{w f_w f_v} + \overline{f_v^2}}{1 + \kappa^2 \overline{w^2}}; \\ E_r(\kappa) &= \frac{\kappa^2 \overline{w^2 f_v^2} + 2\kappa \overline{w f_w f_v} + \overline{f_w^2}}{1 + \kappa^2 \overline{w^2}}. \end{aligned} \qquad (16)$$

A hat ($\hat{\bullet}$) above a quantity indicates an approximation. Since the fit energy E_f might be a function of the coordinate w, as is the adapted weighting function, optimization lead to a false curvature estimate. Therefore, minimization of the residual energy E_r is used to find the following closed-form curvature estimate:

$$\hat{\kappa} = \frac{\overline{w^2 f_v^2} - \overline{w^2} \cdot \overline{f_w^2} - \sqrt{4\overline{w^2} \cdot \overline{w f_w f_v}^2 + \left(-\overline{w^2 f_v^2} + \overline{w^2} \cdot \overline{f_w^2}\right)^2}}{2\overline{w^2} \cdot \overline{w f_w f_v}}. \qquad (17)$$

The confidence measure can now be computed in two different ways. The confidence measure $c(\phi, \kappa)$ has its maximum at $\hat{\phi}_{opt}, \hat{\kappa}_{opt}$. To avoid an interative search for this optimum, one can compute $c(\hat{\phi}, \hat{\kappa})$ by subsituting $\hat{\phi}$ and $\hat{\kappa}$ in (15) and (4). Note that estimates $\hat{\phi}$ $\hat{\kappa}$ do not have to be equal to the values of ϕ and κ that optimize the confidence function. However, computing $c(\hat{\phi}, \hat{\kappa})$ is still expensive. A significant speed-up can be obtained by approximating the confidence measure using the approximate energies of (16).

$$\hat{c}(\phi,\kappa) = \frac{\hat{E}_f(\phi,\kappa) - \hat{E}_r(\phi,\kappa)}{\hat{E}_f(\phi,\kappa) + \hat{E}_r(\phi,\kappa)}. \tag{18}$$

Again, one can avoid an iterative search by substituting $\hat{\phi}$ and $\hat{\kappa}$ in (18), which yield $\hat{c}(\hat{\phi},\hat{\kappa})$.

The curvature estimator in (17) is the tangential or isophote curvature. The normal (or gradient flow line) curvature [17] can be computed by exchanging the v and w coordinates in (12) and (13).

5 Implementation

Direct computation of the curvature and the confidence measure is a space-variant operation. This yields a high computational demand. Fortunately, (16) to (18) can be calculated with global convolutions which can be implemented by multiplication in the Fourier-domain. This yields a substantial reduction in computational complexity. The derivatives f_x and f_y are implemented as regularized derivative filters:

$$f_x \equiv f(x,y) \otimes \frac{\partial g(x,y;\sigma_g)}{\partial x} \overset{F}{\longleftrightarrow} j\omega_x \tilde{f}(\omega_x,\omega_y)\tilde{g}(\omega_x,\omega_y;\sigma_g), \tag{19}$$

with $\tilde{f}$ the Fourier transform of f and $g(x,y;\sigma_g)$ a Gaussian regularization function of scale σ_g.

$$g(x,y;\sigma_g) = \frac{1}{2\pi\sigma_g^2} e^{-(x^2+y^2)/2\sigma_g^2} \overset{F}{\longleftrightarrow} \tilde{g}(\omega_x,\omega_y;\sigma_g) = e^{-\frac{1}{2}(\omega_x^2+\omega_y^2)\sigma_g^2}. \tag{20}$$

The terms of the curvature estimator and the confidence measure, (16) and (17) , are expanded in Appendix B (the circular model is treated in Appendix A). The remaining terms $\overline{x^p y^q f_x^r f_y^s}$ are conveniently calculated as multiplications in the Fourier domain

$$\overline{x^p y^q f_x^r f_y^s} = u(p,q,\sigma_a) \otimes f_x^r f_y^s \overset{F}{\longleftrightarrow} \tilde{u}(p,q,\sigma_a) F\left\{f_x^r f_y^s\right\}. \tag{21}$$

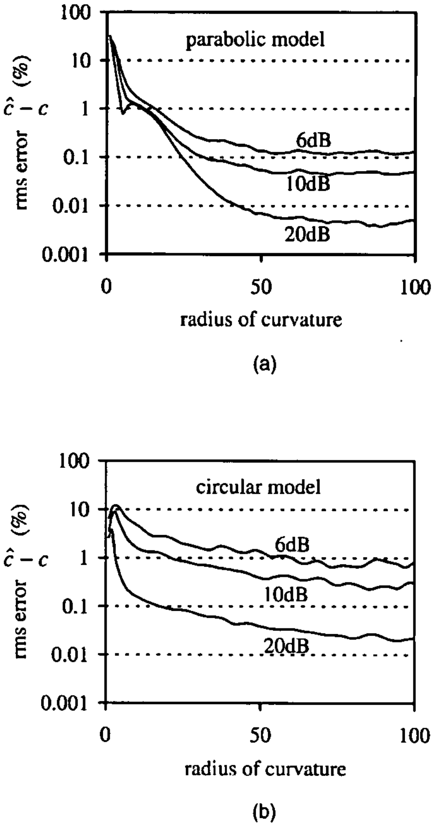
For the window function, we choose a Gaussian of scale σ_a. The Fourier transform of the filter $u()$ is

$$u(p,q,\sigma_a) \equiv x^p y^q g(x,y;\sigma_a) \overset{F}{\longleftrightarrow} \tilde{u}(p,q;\sigma_t) \equiv j^{p+q} \frac{\partial^{p+q}\tilde{g}(\omega_x,\omega_y;\sigma_a)}{\partial\omega_x^p \partial\omega_\eta^q}. \tag{22}$$

Due to the high frequency character of oriented patterns, σ_g should be kept small, i.e., tuned to the frequency characteristics of the cross-section of a line. Noise suppression is accomplished by averaging all terms by Gaussian window (size σ_a), i.e., the size of the curvilinear model.

6 Experiments

In this section the proposed algorithms are tested on synthetic and real-world images. The feature extraction which we presented is based upon finding a maximum of the confidence measure in parameter space c(a). The curvature of oriented patterns corresponds to the position of the maximum in $c(\kappa,\phi)$-space. To avoid searching κ,ϕ-space, the approximations $\hat{\phi}$ and $\hat{\kappa}$ are proposed. With these, an approximated confidence measure $\hat{c}$ and the exact confidence measure c may be computed. The goal of the experiments is to investigate the performance of these approximations as a function of the curvature. Also, the robustness with respect to the noise is checked. The tests are performed on a concentric circle image $f(x,y) = \sin\left(\sqrt{x^2+y^2}+\varphi\right)+n$ (see Fig. 1) in which $n = N(0,\sigma_n^2)$ and φ is a phase-term set randomly for every noise-realization. For the signal-to-noise ratio, we use $SNR = 10\log(h^2/\sigma_n^2)$, where h is the contrast difference and σ_n the standard deviation of the noise. Be aware that the proposed algorithms are based on the gradient energy of the local image. Thus, an increase of the pattern frequency will usually result in a higher SNR (gradient energy versus filtered noise variance) and, therefore, a better performance. All experiments on the concentric circle image are based on 100 measurements. Unless mentioned otherwise the sigma sizes are $\sigma_g = 1.0$ and $\sigma_a = 5.0$.



(a)

(b)

Fig. 3. Rms error between the actual confidence measure and its approximation as a function of the radius. (a) Approximate parabolic model applied to concentric circle patterns of various SNR (top-to-bottom: 6 dB, 10 dB, 20 dB). (b) Approximate circular model applied to concentric circle patterns of various SNR (top-to-bottom: 6 dB, 10 dB, 20 dB).

Confidence measure as selection criterion. The importance of choosing the right model is illustrated in Fig. 1, which shows the confidence measures of the circular, parabolic, and straight model applied to a noise-free pattern of concentric circles. It is clear that, for high curvatures, the deviation of the straight and the parabolic model form the circle pattern results in a significantly lower value of the confidence measure.

Bias of the actual confidence measure. To investigate to what extent the optimum of the confidence function in κ,ϕ-space is found, we compare the average confidence measure of the circular model applied to curved patterns with the average confidence measure of a straight model applied to a straight pattern. Both images have identical signal-to-noise ratios. The confidence measure $c(\phi,\kappa)$ of a curvilinear model can be slightly higher than the confidence measure of a straight model. This slight increase is

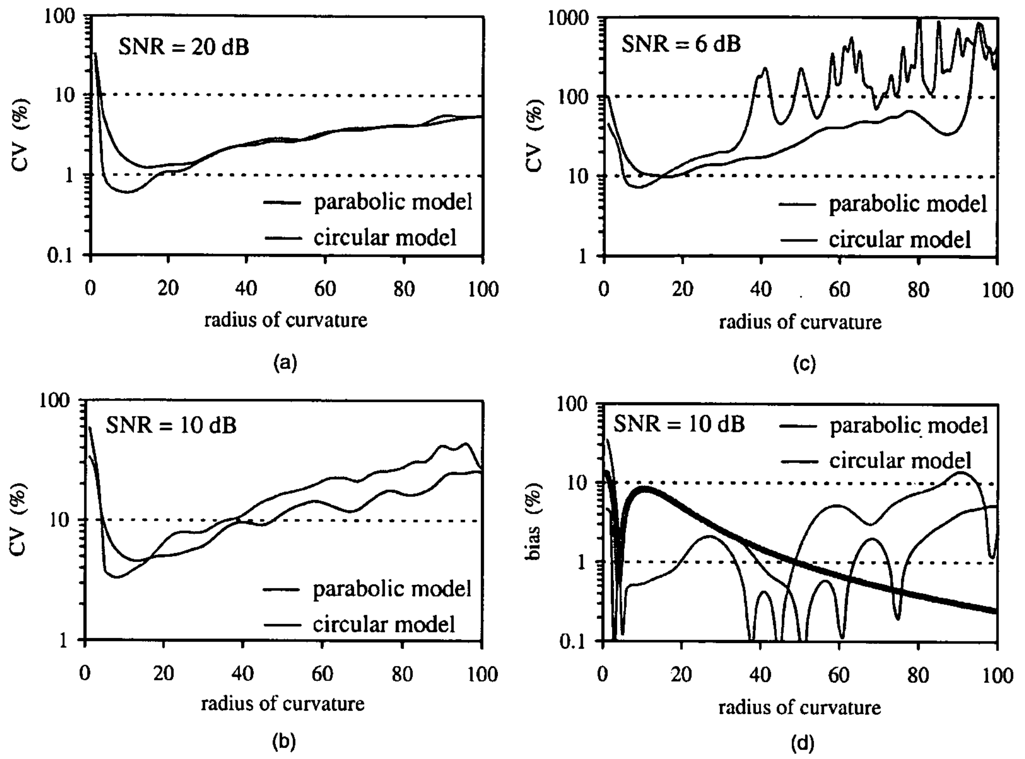


Fig. 4. Curvature estimators using curvilinear models: black line = parabolic model, gray line = circular model. (a), (b) and (c) Coefficient-of-variation (CV) for the parabolic and circular model-based curvature estimators for different SNR (20 dB,10 dB, 6dB). (d) Bias of parabolic and circular model based curvature estimators (SNR = 10 dB) (thick gray line indicates the noise-free bias of curvature using the parabolic model).

caused by the fact that the curved model allows for two parameters to adjust to the noise.

The average confidence measure $c(\hat{\phi}, \hat{\kappa})$ of the circular model applied to the concentric circles is depicted in Fig. 2 for three SNR's (20dB, 10dB, 6dB). It clearly shows that, for small radii, the average confidence measure $c(\hat{\phi}, \hat{\kappa})$, decreases. This is caused by an increasing discrepancy between the approximated $(\hat{\phi}, \hat{\kappa})$ and the optimal $(\phi_{opt}, \kappa_{opt})$ for small radii. Note, $c(\phi_{opt}, \kappa_{opt})$ does not decrease for small radii. Fig. 2b indicates the variation around the average confidence measure for the straight model. Increasing the window size (local image) reduces the variation in exchange of a further decrease of $c(\hat{\phi}, \hat{\kappa})$ for small radii.

Approximation error of the confidence measure. In Section 4, we presented two methods for computing the confidence measure, the actual confidence measure $c(\hat{\phi}, \hat{\kappa})$ and an approximation $\tilde{c}(\hat{\phi}, \hat{\kappa})$. In Fig. 3, the rms (root-mean-square) error due to this approximation is depicted for the circular and the parabolic model. For both models, these errors are small. Only for high curvatures (small radii), it may be worthwhile to compute the actual confidence measure.

Robustness of the curvature estimator. It is important to test the robustness of the curvature estimation. In Fig. 4, the noise sensitivity of the parabolic and circular curvature estimators is depicted. Both models were applied to the concentric circles. The coefficient-of-variation ($CV = \sigma/\mu$) of both models is similar for the middle and high SNRs, but the parabolic models performs better for low SNRs. Considering the advantage of the circular curvature estimator due to the exact match between the model and the pattern, we show that the parabolic curvature estimator suffers less from the approximations. The parabolic curvature estimator performs at least as well over a wide range of curvatures. Only for high curvatures, the circular model can take advantage of the exact match. In practice, one can compute the curvature corresponding to both models. The one with the highest confidence measure is preferred because its model yields a better description of the data.

Application of curvilinear models to real-world data sets. In Fig. 5, an interference pattern, together with the curvature and confidence estimation for both the parabolic and circular model, is depicted. As expected, the parabolic model fails in the middle of the ellipses, as indicated by an abrupt drop of the confidence measure. The circular confidence measure hardly decreases for the circles at the top and the bottom of the image. For the flatter ellipses on the left and the right, the mismatch between the model and the pattern is slightly larger. In the difference image between the circular and parabolic confidence measures, the lighter areas indicate a better description of the circular model, whereas, in the darker areas, the parabolic model yields a better fit. The slightly darker lines denote an almost perfect parabolic line pattern. The isophote curvature fails at the ridges (dark lines) and valleys (white lines) as expected (see Section 1).

The estimated local curvature of a fingerprint and a CT cross-section of a tree-trunk are depicted in Fig. 6. Both curvilinear models produced similar results. The dark lines in the logarithmically stretched curvature images denote locally straight patterns. Both peaks in the fingerprints curvature correspond to important minutia for fingerprint recognition [15], [16]. The isophote

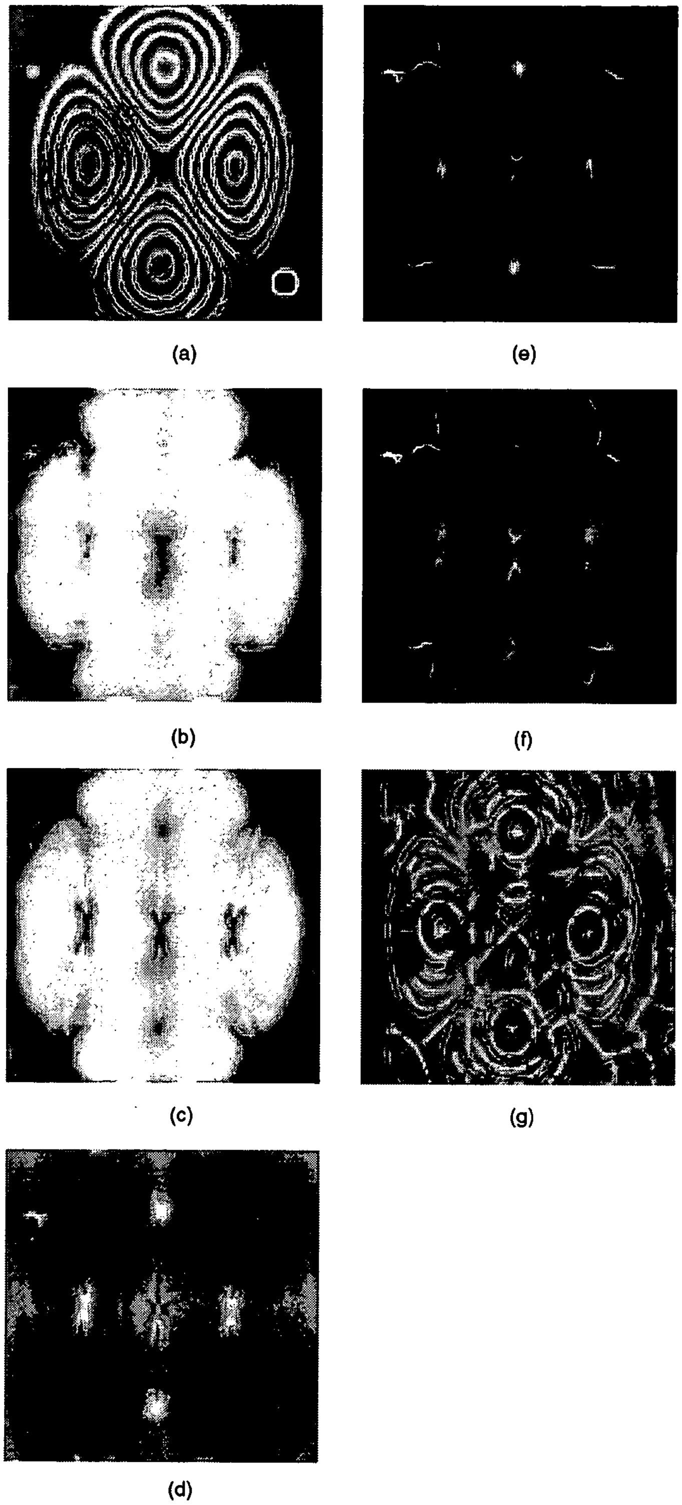


Fig. 5. (a) Interference pattern of a vibrating plate. The superimposed circle denotes the size of the curvilinear model. (b) and (c) Confidence measures for, respectively, the circular and parabolic model (range [0,1]) computed with $\sigma_g = 1.0$ and $\sigma_a = 5.0$. (d) Difference in confidence measure between circular and parabolic model (range [-0.5, 0.5]). (e) and (f) Estimated curvatures $\hat{\kappa}$ for respectively the circular and parabolic model (log stretched). (g) Isophote curvature at scale $\sigma = 5$ (range [-1,1]).

curvature fails at the ridges (dark lines) and valleys (white lines) as expected (see Section 1). The curvilinear curvature can be used to improve (to prevent jumping the rails) the ridge tracking [16], which is already based on orientation estimation. The high confidence measures (white areas in confidence images) indicate a perfect fit of the model and a reliable curvature estimate.

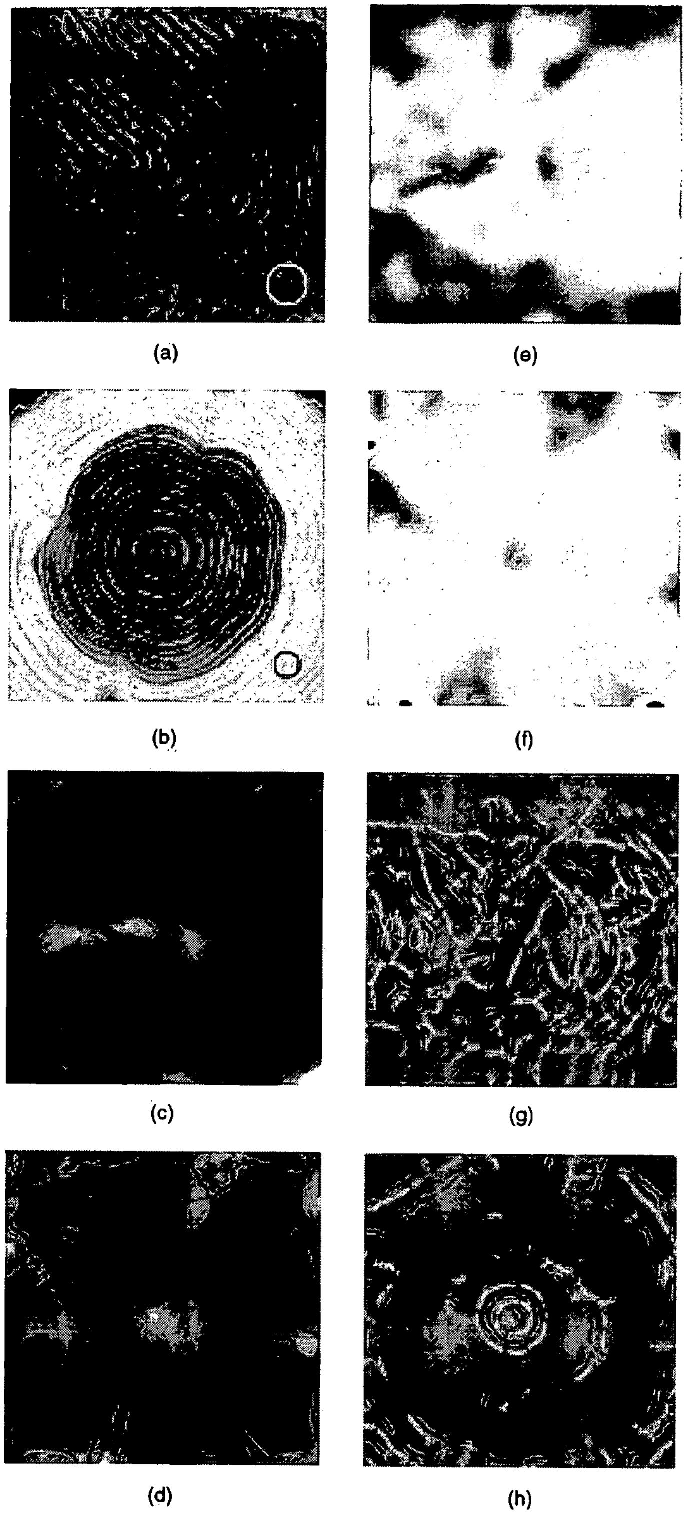


Fig. 6. (a) Fingerprint image. (b) CT image of trunk. (c) and (d) The estimated curvature $\hat{\kappa}$ using the parabolic model (log stretched) at $\sigma_g = 1.0$ and $\sigma_a = 5.0$. (e) and (f) The confidence measure of the parabolic model (range [0, 1]), (g) and (h) Isophote curvature at scale $\sigma = 5$ (range [-1, 1]).

7 Conclusions

In this paper, we present a method to compare a local image with a model function. A quality measure indicates the resemblance between the local image and the model function. Feature extraction is obtained by optimization of the quality function as a function of the parameters which represent the feature. The quality function is interpreted as a confidence measure for the measured features. We propose two curvilinear models to describe curved oriented patterns. To avoid searching ϕ, κ-space, we propose, closed-form solution for approximations to the actual parameters of the curvilinear models $\hat{\phi}$ and $\hat{\kappa}$. Instead of the exact confidence measure $c(\hat{\phi}, \hat{\kappa})$ an approximation $\hat{c}(\hat{\phi}, \hat{\kappa})$, can be computed resulting in a huge reduction in computational demand. We demonstrate that these approximations yield good results for almost all curvatures. Only for the highest curvatures, one might decide to compute $c(\hat{\phi}, \hat{\kappa})$ or (even more computationally demanding) to iterate in ϕ, κ-space for $c(\phi_{opt}, \kappa_{opt})$.

Appendix A

For a concentric circle model, $u(x, y, \phi, \kappa) = \sqrt{\kappa^2 w^2 + (1 - \kappa v)^2}$, the fit and residual energies are

$$
\begin{aligned}
E_f(\kappa) &= \overline{\left(\frac{(1-\kappa v)^2 f_v^2 - 2\kappa w(1-\kappa v) f_v f_w + \kappa^2 w^2 f_w^2}{(1-\kappa v)^2 + \kappa^2 w^2}\right)} \\
E_r(\kappa) &= \overline{\left(\frac{(1-\kappa v)^2 f_w^2 + 2\kappa w(1-\kappa v) f_v f_w + \kappa^2 w^2 f_v^2}{(1-\kappa v)^2 + \kappa^2 w^2}\right)}.
\end{aligned}
\tag{23}
$$

To obtain a closed-form solution for the curvature and the confidence measure, the local energies are computed inside a $\left(\kappa^2 w^2 + (1 - \kappa v)^2\right)$-weighted space-variant window. This yields

$$
\begin{aligned}
\hat{E}_f &= \frac{\kappa^2\left(\overline{v^2 f_v^2 + 2vw f_v f_w + w^2 f_w^2}\right) + 2\kappa\left(\overline{-v f_v^2 - w f_v f_w}\right) + \overline{f_v^2}}{1 - 2\kappa\bar{v} + \kappa^2\left(\overline{v^2 + w^2}\right)} \\
&\equiv \frac{A\kappa^2 + 2B\kappa + C}{1 + D\kappa^2} \\
\hat{E}_r &= \frac{\kappa^2\left(\overline{v^2 f_w^2 - 2vw f_v f_w + w^2 f_v^2}\right) + 2\kappa\left(\overline{-v f_w^2 + w f_v f_w}\right) + \overline{f_w^2}}{1 - 2\kappa\bar{v} + \kappa^2\left(\overline{v^2 + w^2}\right)} \\
&\equiv \frac{E\kappa^2 + 2F\kappa + G}{1 + D\kappa^2},
\end{aligned}
\tag{24}
$$

with $\bar{v} = 0$. The minimization of the residual energy yields an approximation of the curvature:

$$
\hat{\kappa} = \frac{E - GD - \sqrt{4F^2 D + (-E + GD)^2}}{2FD}. \tag{25}
$$

The terms of $\hat{E}_f$ and$\hat{E}_g$ are expanded with (14) and

$$
f_v = f_x \cos\phi + f_y \sin\phi \qquad f_w = -f_x \sin\phi + f_y \cos\phi. \tag{26}
$$

This results in

$$
\begin{cases}
A = \overline{x^2 f_x^2} + 2\overline{xy f_x f_y} + \overline{y^2 f_y^2} \\
B = -\left(\overline{x f_x^2} + \overline{y f_x f_y}\right)\cos\phi - \left(\overline{x f_x f_y} + \overline{y f_y^2}\right)\sin\phi \\
C = \overline{f_x^2}\cos^2\phi + 2\overline{f_x f_y}\cos\phi\sin\phi + \overline{f_y^2}\sin^2\phi \\
D = 2\sigma_a^2 \\
E = \overline{x^2 f_y^2} - 2\overline{xy f_x f_y} + \overline{y^2 f_x^2} \\
F = \left(\overline{y f_x f_y} - \overline{x f_y^2}\right)\cos\phi + \left(\overline{x f_x f_y} - \overline{y f_x^2}\right)\sin\phi \\
G = \overline{f_y^2}\cos^2\phi - 2\overline{f_x f_y}\cos\phi\sin\phi + \overline{f_x^2}\sin^2\phi.
\end{cases}
\tag{27}
$$

The averaged terms can be computed as global convolutions (see Section 5). The approximated confidence function is computed with

$$
\hat{c} = \frac{\kappa^2(A - E) + 2\kappa(B - F) + (C - G)}{\kappa^2(A + E) + 2\kappa(B + F) + (C + G)}. \tag{28}
$$

Appendix B

The terms for the parabolic confidence measure (16) and curvature estimator (17) are

$$
\begin{cases}
\overline{w^2 f_w^2} = \overline{y^2 f_y^2}\cos^4\phi - 2\left(\overline{xy f_y^2} + \overline{y^2 f_x f_y}\right)\cos^3\phi\sin\phi \\
+\left(\overline{x^2 f_y^2} + \overline{4xy f_x f_y} + \overline{y^2 f_x^2}\right)\cos^2\phi\sin^2\phi \\
+2\left(-\overline{x^2 f_x f_y} - \overline{xy f_x^2}\right)\cos\phi\sin^3\phi + \overline{x^2 f_x^2}\sin^4\phi \\
\overline{w^2 f_v^2} = \overline{y^2 f_x^2}\cos^4\phi - 2\left(\overline{xy f_x^2} - \overline{y^2 f_x f_y}\right)\cos^3\phi\sin\phi \\
+\left(\overline{x^2 f_x^2} - \overline{4xy f_x f_y} + \overline{y^2 f_y^2}\right)\cos^2\phi\sin^2\phi \\
+2\left(\overline{x^2 f_x f_y} - \overline{2xy f_y^2}\right)\cos\phi\sin^3\phi + \overline{x^2 f_y^2}\sin^4\phi \\
\overline{w f_v f_w} = \overline{y f_x f_y}\cos^3\phi + \left(-\overline{x f_x f_y} - \overline{y\left(f_x^2 - f_y^2\right)}\right)\cos^2\phi\sin\phi \\
+\left(\overline{x\left(f_x^2 - f_y^2\right)} - \overline{y f_x f_y}\right)\cos\phi\sin^2\phi + \overline{x f_x f_y}\sin^3\phi \\
\overline{w^2} = \sigma_a^2
\end{cases}
\tag{29}
$$

for $\overline{f_v^2}$ and $\overline{f_w^2}$, see term, C and G in Appendix A.

Acknowledgments

The research was performed while J. van de Weijer was with the Delft University of Technology. This work was partially supported by the Royal Netherlands Academy of Arts and Sciences (KNAW) and the Rolling Grants program of the Foundation for Fundamental Research in Matter (FOM).

References

[1] M. Kass and A. Witkin, "Analyzing Oriented Patterns," *Computer Vision, Graphics, and Image Processing*, vol. 37, pp. 362-385, 1987.

[2] J. Bigun, G.H. Granlund, and J. Wiklund, "Multidimensional Orientation Estimation with Applications to Texture Analysis and Optical Flow," *IEEE Trans. Pattern Analysis and Machine Intelligence*, vol. 13, pp. 775-790, 1991.

[3] L. Haglund, "Adaptive Multi-Dimensional Filtering," Linkoping Univ., Sweden, 1992.

[4] L.J. van Vliet and P.W. Verbeek, "Estimators for Orientation and Anisotropy in Digitized Images," *Proc. First Conf. Advanced School for Computing and Imaging*, pp. 442-450, 1995.

[5] P.W. Verbeek, L.J. van Vliet, and J. van de Weijer, "Improved Curvature and Anisotropy Estimation for Curved Line Bundles," *Proc. 14th Int'l Conf. Pattern Recognition*, pp. 528-533, 1998.

[6] M. Worring and A.W.M. Smeulders, "Digital Curvature Estimation," *CVGIP: Image Understanding*, vol. 58, pp. 366-382, 1993.

[7] P.W. Verbeek, "A Class of Sampling-Error Free Measures in Oversampled Band-Limited Images," *Pattern Recognition Letters*, vol. 3, pp. 287-292, 1985.

[8] L.J. van Vliet and P.W. Verbeek, "Curvature and Bending Energy in 2D and 3D Digitized Images," *Proc. Eighth Scandinavian Conf. Image Processing, SCIA '93*, pp. 1403-1410, 1993.

[9] M. van Ginkel, P.W. Verbeek, and L.J. van Vliet, "Curvature Estimation for Overlapping Curved Patterns Using Orientation Space," *Proc. Fourth Conf. Advanced School for Computing and Imaging*, pp. 173-178, 1998.

[10] M. van Ginkel, J. van de Weijer, L.J. van Vliet, and P.W. Verbeek, "Curvature Estimation from Orientation Fields," *Proc. 11th Scandinavian Conf. Image Analysis, SCIA '99*, 1999.

[11] J. Hansen and J. Bigun, "Local Symmetry Modeling Multi-Dimensional Images," *Pattern Recognition Letters*, vol. 13, pp. 253-262, 1992.

[12] J. Bigun and M.H. du Buf, "Symmetry Interpretation of Complex Moments and the Local Power Spectrum," *J. Visual Comm. and Image Representation*, vol. 6, pp. 154-163, 1995.

[13] J. Bigun, "Pattern Recognition in Images by Symmetry and Coordinate Transformations," *Computer Vision and Image Understanding*, vol. 68, pp. 290-307, 1997.

[14] H. Knutsson and C.-F. Westin, "Normalised Convolution: A Technique for Filtering Incomplete and Uncertain Data," *Proc. Eighth Scandinavian Conf. Image Processing, SCIA '93*, 1993.

[15] D. Maio and D. Maltoni, "Direct Gray-Scale Minutiae Detection in Fingerprints," *IEEE Trans. Pattern Analysis and Machine Intelligence*, vol. 19, pp. 27-40, 1997.

[16] A. Jain, L. Hong, and R. Bolle, "On-Line Fingerprint Verification," *IEEE Trans. Pattern Analysis and Machine Intelligence*, vol. 19, pp. 302-314, 1997.

[17] P. Breton and S.W. Zucker, "Shadows and Shading Flow Fields," *Proc. IEEE Conf. Computer Vision and Pattern Recognition CVPR '96*, pp. 782-789, 1996.

WOLFRAMRESEARCH mathworld.wolfram.com OTHER WOLFRAM SITES ▸

Search Site Go

mathworld

INDEX
Algebra
Applied Mathematics
Calculus and Analysis
Discrete Mathematics
Foundations of Mathematics
Geometry
History and Terminology
Number Theory
Probability and Statistics
Recreational Mathematics
Topology

Alphabetical Index

ABOUT THIS SITE
About *MathWorld*
About the Author
Terms of Use

DESTINATIONS
What's New
Headline News (RSS)
Random Entry
Animations
Live 3D Graphics

CONTACT
Email Comments
Contribute!
Sign the Guestbook

MATHWORLD - IN PRINT
Order book from Amazon

Calculus and Analysis ▸ Differential Geometry ▸ Differential Geometry of Surfaces ▾

Normal Curvature

Let $\mathbf{u_p}$ be a unit tangent vector of a regular surface $M \subset \mathbb{R}^3$. Then the normal curvature of *M* in the direction $\mathbf{u_p}$ is

$$\kappa(\mathbf{u_p}) = S(\mathbf{u_p}) \cdot \mathbf{u_p}, \qquad (1)$$

where *S* is the shape operator. Let $M \subset \mathbb{R}^3$ be a regular surface, $\mathbf{p} \in M$, **x** be an injective regular patch of *M* with $\mathbf{p} = \mathbf{x}(u_0, v_0)$, and

$$\mathbf{v_p} = a\mathbf{x}_u(u_0, v_0) + b\mathbf{x}_v(u_0, v_0), \qquad (2)$$

where $\mathbf{v_p} \in M_\mathbf{p}$. Then the normal curvature in the direction $\mathbf{v_p}$ is

$$\kappa(vp) = \frac{ea^2 + 2fab + gb^2}{Ea^2 + 2Fab + Gb^2}, \qquad (3)$$

where *E*, *F*, and *G* are the coefficients of the first fundamental form and *e*, *f*, and *g* are the coefficients of the second fundamental form.

The maximum and minimum values of the normal curvature at a point on a regular surface are called the principal curvatures κ_1 and κ_2.

SEE ALSO: Curvature, Fundamental Forms, Gaussian Curvature, Mean Curvature, Principal Curvatures, Shape Operator, Tangent Vector

PAGES LINKING HERE: search

REFERENCES:

Euler, L. "Recherches sur la courbure des surfaces." *Mém. de l'Acad. des Sciences, Berlin* **16**, 119-143, 1760.

Gray, A. "Normal Curvature." §18.2 in *Modern Differential Geometry of Curves and Surfaces with Mathematica, 2nd ed.* Boca Raton, FL: CRC Press, pp. 363-367, 1997.

Meusnier, J. B. "Mémoire sur la courbure des surfaces." *Mém. des savans étrangers* **10** (lu 1776), 477-510, 1785.

CITE THIS AS:

Eric W. Weisstein. "Normal Curvature." From *MathWorld*--A Wolfram Web Resource. http://mathworld.wolfram.com/NormalCurvature.html

WOLFRAMRESEARCH mathworld.wolfram.com OTHER WOLFRAM SITES ▸
Search Site GO

mathworld

INDEX
Algebra
Applied Mathematics
Calculus and Analysis
Discrete Mathematics
Foundations of Mathematics
Geometry
History and Terminology
Number Theory
Probability and Statistics
Recreational Mathematics
Topology

Alphabetical Index

ABOUT THIS SITE
About *MathWorld*
About the Author
Terms of Use

DESTINATIONS
What's New
Headline News (RSS)
Random Entry
Animations
Live 3D Graphics

CONTACT
Email Comments
Contribute!
Sign the Guestbook

MATHWORLD - IN PRINT
Order book from Amazon

Calculus and Analysis ▸ Differential Geometry ▸ Differential Geometry of Surfaces ▾

Shape Operator

The negative derivative

$$S(\mathbf{v}) = -D_{\mathbf{v}}\mathbf{N} \tag{1}$$

of the unit normal **N** vector field of a surface is called the shape operator (or Weingarten map or second fundamental tensor). The shape operator S is an extrinsic curvature, and the Gaussian curvature is given by the determinant of S. If $\mathbf{x} : U \to \mathbb{R}^3$ is a regular patch, then

$$S(\mathbf{x}_u) = -\mathbf{N}_u \tag{2}$$

$$S(\mathbf{x}_v) = -\mathbf{N}_v. \tag{3}$$

At each point **p** on a regular surface $M \subset \mathbb{R}^3$, the shape operator is a linear map

$$S : M_{\mathbf{p}} \to M_{\mathbf{p}}. \tag{4}$$

The shape operator for a surface is given by the Weingarten equations.

SEE ALSO: Curvature, Fundamental Forms, Weingarten Equations

PAGES LINKING HERE: search

REFERENCES:

Gray, A. "The Shape Operator," "Calculation of the Shape Operator," and "The Eigenvalues of the Shape Operator." §16.1, 16.3, and 16.4 in *Modern Differential Geometry of Curves and Surfaces with Mathematica, 2nd ed.* Boca Raton, FL: CRC Press, pp. 360-363 and 367-372, 1997.

Reckziegel, H. In *Mathematical Models from the Collections of Universities and Museums* (Ed. G. Fischer). Braunschweig, Germany: Vieweg, p. 30, 1986.

CITE THIS AS:

Eric W. Weisstein. "Shape Operator." From *MathWorld*--A Wolfram Web Resource. http://mathworld.wolfram.com/ShapeOperator.html

On-line Fingerprint Verification

Anil Jain and Lin Hong
Pattern Recognition and Image Processing Laboratory
Department of Computer Science
Michigan State University
East Lansing, MI 48824, USA
{jain,honglin}@cps.msu.edu

Abstract

We describe the design and implementation of an on-line fingerprint verification system which operates in two stages: (i) minutia extraction and (ii) minutia matching. An improved minutia extraction algorithm that is much faster and more accurate than our earlier algorithm [8] has been implemented. For minutia matching, an alignment-based elastic matching algorithm has been developed. This algorithm is capable offinding the correspondences between input minutiae and the stored template without resorting to exhaustive search and has the ability to adaptively compensate for the nonlinear deformations and inexact pose transformations between fingerprints. The system has been tested on two sets of fingerprint images captured with inkless scanners. The verification accuracy is found to be over 99% with a 15% reject rate. Typically, a complete fingerprint verification procedure takes, on an average, about 8 seconds on a SPARC 20 workstation. It meets the response time requirements of on-line verification with high accuracy.

1. Introduction

Fingerprint verification is one of the most reliable personal identification methods [2, 4]. It plays a very important role in forensic and civilian applications such as criminal identification, access control, and ATM card verification [4]. However, manual fingerprint verification is so tedious, time-consuming, and expensive that it is incapable of meeting today's increasing performance requirements. As a result, automatic fingerprint identification systems (AFIS) are in great demand [4].

In this paper, we will introduce an *on-line fingerprint verification system* which is capable of verifying identities of people in "real time". Such a system has great utility in a variety of personal identification and access control applications. The overall block diagram of our system is shown in Figure 1. It operates as follows: *(i) off-line phase*: a digital image of one fingerprint of a person to be verified is captured; a feature extraction algorithm is applied; minutiae are extracted and stored as a template for later use; *(ii) on-line phase*: the individual to be verified indicates his/her identity and places his/her finger on the inkless scanner; an digital image of his/her fingerprint is captured; minutiae are extracted from the captured image and fed to a matching algorithm, which matches it against the stored template.

Figure 1. Overview of the system

In the following sections, we will describe in detail our on-line fingerprint verification system. Section 2 mainly discusses the feature extraction algorithm. Section 3 presents our minutia matching algorithm. Experimental results on fingerprint databases captured with inkless scanners are described in section 4. Section 5 contains the summary and discussion.

2. Minutia Extraction

We have implemented a minutia detection algorithm which is a modified version of the technique proposed by Ratha *et al.* [8]. The overall flowchart is depicted in Figure 2.

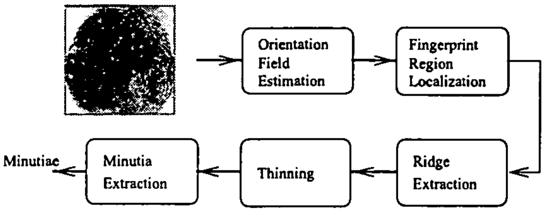


Figure 2. Flowchart of the minutia extraction algorithm

2.1. Estimation of Orientation Field

A new hierarchical implementation of Rao's algorithm [7] is used to estimate the orientation field of an input fingerprint image. It consists of the following two main steps:

1. Estimate the local orientation at each pixel using following formula:

$$\theta_o = \frac{1}{2}tan^{-1}(\frac{\sum_{i=1}^{W}\sum_{j=1}^{W} 2G_x(i,j)G_y(i,j)}{\sum_{i=1}^{W}\sum_{j=1}^{W}(G_x^2(i,j) - G_y^2(i,j))}), \quad (1)$$

   where W is the size of the local window; G_x and G_y are the gradient magnitudes in x and y directions, respectively.

2. Compute the *variance* of the orientation field in a local neighborhood at each pixel (i,j). If it is above a certain threshold T_v, then the local orientation at this pixel is re-estimated at a lower resolution level until it is above the threshold value.

Experimental results show that even in the presence of noise, smudges, and breaks in ridges, a fairly smooth orientation field estimate can be obtained with this algorithm. Before the ridge detection, a segmentation algorithm which bases its judgment on the local variance of grey levels is used to locate the fingerprint region in the image.

2.2. Ridge Detection

The most salient property corresponding to ridges in a fingerprint image is that grey level values on ridges attain their local maxima along the normal directions of local ridges. In our minutia detection algorithm, a fingerprint image is first convolved with the following two masks, $h_t(x,y;u,v)$ and $h_b(x,y;u,v)$ of size $W \times H$, which are capable of adaptively accentuating the local maximum grey level values along the normal direction of the local ridge directions:

$$h_t(x,y;u,v) = \begin{cases} \frac{-1}{\sqrt{2\pi}\delta}e^{\frac{-u}{\delta^2}}, & \text{if } u = l(v) - d, v \in \Omega \\ \frac{1}{\sqrt{2\pi}\delta}e^{\frac{-u}{\delta^2}}, & \text{if } u = l(v), v \in \Omega \\ 0, & \text{otherwise,} \end{cases} \quad (2)$$

$$h_b(x,y;u,v) = \begin{cases} \frac{-1}{\sqrt{2\pi}\delta}e^{\frac{-u}{\delta^2}}, & \text{if } u = l(v) + d, v \in \Omega \\ \frac{1}{\sqrt{2\pi}\delta}e^{\frac{-u}{\delta^2}}, & \text{if } u = l(v), v \in \Omega \\ 0, & \text{otherwise,} \end{cases} \quad (3)$$

$$l(v) = v\tan(\theta(x,y)), \quad (4)$$

$$d = \frac{H}{2\cos(\theta(x,y))}, \quad (5)$$

$$\Omega = H\left[\left|\frac{\sin(\theta(x,y))}{-2}\right|, \left|\frac{\sin(\theta(x,y))}{2}\right|\right], \quad (6)$$

where $\theta(x,y)$ represents the local ridge orientation at pixel (x,y). If *both* the grey level values at pixel (x,y) of the convolved images are larger than a certain threshold T_r, then pixel (x,y) is labeled as a ridge. By adapting the mask width to the width of the local ridge, this algorithm can efficiently locate the ridges in a fingerprint image. After the above steps, a thinning algorithm is applied to obtain a thinned 8-connected ridge map.

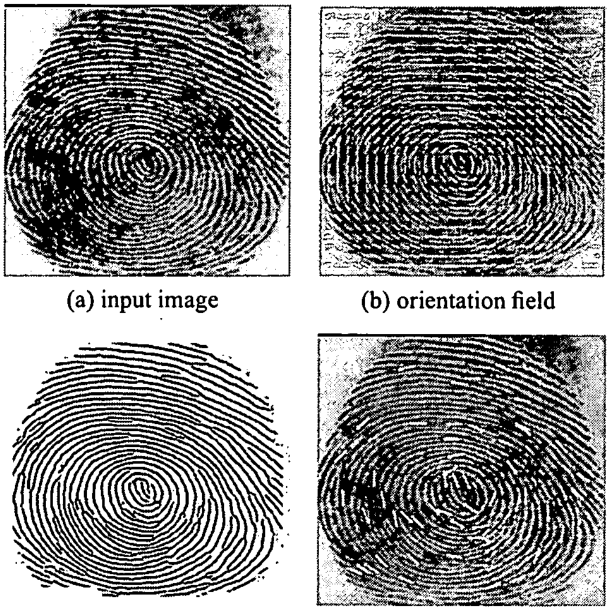
2.3. Minutia Detection

Without a loss of generality, we can assume that if a pixel is on a thinned ridge (8-connected), then it has a value 1, and 0 otherwise. Let $N_0, N_1, ..., N_7$ denote the 8 neighbors of a given pixel (x,y), then pixel (x,y) is a ridge ending if $\sum_{i=0}^{8} N_i = 1$ and a ridge bifurcation if $\sum_{i=0}^{8} N_i > 2$. However, the presence of undesired spikes and breaks present in a thinned ridge map may lead to many spurious minutiae being detected. Therefore, the following heuristics are used in preprocessing: (i) If a branch in a ridge map is orthogonal to the local ridge directions and its length is less than a specified threshold T_b, then it will be removed; (ii) If a break in a ridge is short enough and no other ridges pass through it, then it will be connected.

For each minutia, the following parameters are recorded: *(i) x-coordinate, (ii) y-coordinate, (iii) orientation, and (iv) the ridge on which it resides.* The recorded ridges are normalized by average local ridge distance along the normal direction of local ridges. Figure 3 shows the results of our minutia extraction algorithm on a fingerprint image captured with an inkless scanner.

3. Minutia Matching

Generally, an automatic fingerprint verification is achieved with minutia matching (point pattern matching) instead of a pixel-wise matching or a ridge pattern



(a) input image (b) orientation field

(d) ridge map (f) extracted minutiae

Figure 3. Results of the minutia extraction algorithm on a fingerprint image (640 × 480) captured with an inkless scanner.

matching of fingerprint images. Because a general point matching problem is essentially intractable, features associated with each point and their relative positions are widely used in the point pattern matching algorithms to reduce the exponential number of search paths [3, 1, 6, 9]. However, these algorithms are inherently slow and are unsuitable for an on-line fingerprint verification system. In our system, an alignment-based matching algorithm is implemented, which decomposes the minutia matching into two stages: (i) *alignment stage*, where transformations such as translation, rotation and scaling between an input pattern and a template are first estimated; the input patterns are then aligned with the template according to the estimated parameters; and (ii) *matching stage*, where both the input pattern and the template are converted to polygons in polar space and an elastic string matching algorithm is used to match the resulting polygons.

Let $P = ((x_1^P, y_1^P, \theta_1^P)^T, ..., (x_M^P, y_M^P, \theta_M^P)^T)$ and $Q = ((x_1^Q, y_1^Q, \theta_1^Q)^T, ..., (x_M^Q, y_N^Q, \theta_N^Q)^T)$ denote the M minutiae in the template and the N minutiae in the input image, respectively. The steps of our alignment-based matching algorithm are given below:

1. Match the ridge associated with each input minutia against the ridge associated with each template minutia and align the two patterns according to the matching result.

2. Convert the representations of template and input minutiae into the polar coordinate representations with respect to the corresponding minutia on which alignment is performed and represent them as two symbolic strings by concatenating each minutia in an increasing order of radial angles:

$$P_p = ((r_1^P, e_1^P, \theta_1^P)^T, ..., (r_M^P, e_M^P, \theta_M^P)^T) \quad (7)$$

$$Q_p = ((r_1^Q, e_1^Q, \theta_1^Q)^T, ..., (r_N^Q, e_N^Q, \theta_N^Q)^T), \quad (8)$$

   where r_*, e_*, and θ_* represent the corresponding radius, radial angle, and normalized minutia orientation with respect to the reference minutia, respectively.

3. Match the resulting strings P_p and Q_p with a modified dynamic-programming algorithm described below to find the 'edit distance' between P_p and Q_p.

4. Compute the matching score of the template and input minutiae as the minimum 'edit distance'.

$$M_{pq} = \frac{100 N_{pair}}{\max\{M, N\}}, \quad (9)$$

   where N_{pair} is the number of minutia pairs which fall within a given bounding box.

The maximum and minimum values of the matching score are 100 and 1, respectively. The former value indicates a perfect match, while the later value indicates no match at all.

3.1. Alignment of Point Patterns

It is well known that corresponding curve segments are capable of aligning two point patterns with high accuracy in the presence of noise and deformations. Each minutia in a fingerprint is associated with a ridge. A true alignment can be achieved by matching and aligning the corresponding ridges (see Figure 4). By matching the corresponding normalized ridges, the relative pose transformation between the input minutiae and the template can be estimated. With the estimated pose transformation, the input minutiae can then be translated and rotated to align the template minutiae.

3.2. Aligned Point Pattern Matching

If two identical point patterns are exactly aligned, each pair of corresponding points are completely overlapping. In such a case, a point pattern matching can be simply achieved by counting the number of overlapping pairs. However, in practice, such a situation is rarely encountered.

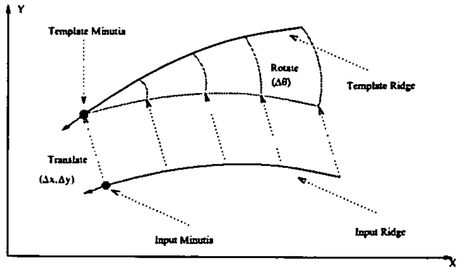


Figure 4. Alignment of the input ridge and the template ridge.

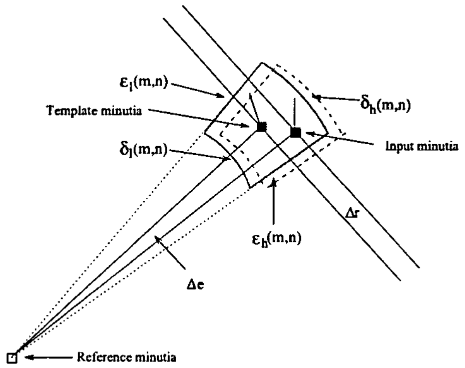


Figure 5. Bounding box and its adjustment.

The error in determining and localizing minutia hinders the alignment algorithm to recover the relative pose transformation exactly. Also, in the presence of nonlinear deformation of fingerprints which is an inherent nature of fingerprint impressions, it is impossible to exactly recover the position of each minutia with respect to its corresponding minutia in the template. Therefore, the aligned point pattern matching algorithm needs to be able to tolerate, to some extent, the deformations due to inexact extraction of minutia positions and nonlinear deformations.

The symbolic string generated by concatenating points in an increasing order of radial angle uniquely represents a point pattern. A modified string matching algorithm which incorporates an elastic term is capable of achieving a certain type of tolerance. However, this algorithm can only tolerate, but not compensate, the adverse effect on the matching produced by the inexact localization of minutia and nonlinear deformations. Therefore, an adaptive mechanism is needed, which should be able to track the local nonlinear deformations and inexact alignment and try to alleviate them during the minimization process. In our string matching algorithm, this adaptation is achieved by adjusting the bounding box (Figure 5) when an inexact match is found during the matching process. It can be represented as follows:

$$\delta_l(m+1,n+1) = \delta_l(m,n) + \eta\Delta r_a, \qquad (10)$$

$$\delta_h(m+1,n+1) = \delta_h(m,n) + \eta\Delta r_a, \qquad (11)$$

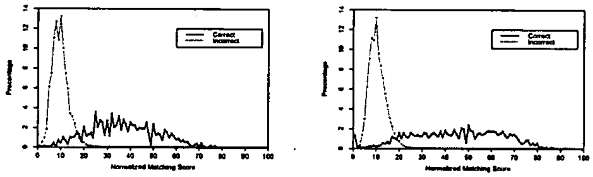
$$\epsilon_l(m+1,n+1) = \epsilon_l(m,n) + \eta\Delta e_a, \qquad (12)$$

$$\epsilon_h(m+1,n+1) = \epsilon_h(m,n) + \eta\Delta e_a, \qquad (13)$$

where $\delta_l(m,n)$, $\delta_h(m,n)$, $\epsilon_l(m,n)$, and $\epsilon_h(m,n)$ specify the adaptive bounding box in radius and radial angle; Δr_a and Δe_a represent the current difference of input and template in radius and radial angle, respectively; η is the learning rate. In our system, the initial values of the bounding box are specified as follows: $\delta_l(0,0) = -8$; $\delta_h(0,0) = +8$; $\epsilon_l(0,0) = -4$; $\epsilon_h(0,0) = +4$. The learning rate, η, is 0.5.

4. Experimental Results

We have tested our system on two sets of fingerprint images. Set 1 contains 10 images (380 × 380) per finger from 18 individuals for a total of 180 images, which were captured with a scanner manufactured by Identix. Set 2 contains 10 images (640 × 480) per finger from 30 individuals for a total of 300 images, which were captured with a scanner manufactured by Digital Biometrics. The captured fingerprint images vary in quality. Approximately 90% are of satisfactory quality, while about 10% are of poor quality.



(a) Identix (b) Digital Biometrics

Figure 6. Distributions of correct and incorrect matching scores: vertical axis represents distribution of matching scores in percentage.

4.1. Matching

Each fingerprint in the test set was matched with the other fingerprints in the set. A matching was labeled correct if the matched fingerprint was among the 9 other fingerprints of the same individual, and incorrect otherwise. The distributions of matching scores are shown in Figure 6. This result indicates that our algorithm is capable of effectively differentiating fingerprints by setting an appropriate value of the matching threshold. Table 1 shows the recognition rates and reject rates with different threshold values on the matching score. We have observed that the incorrect matchings occur mainly due to fingerprint images with poor quality.

Threshold Value	Recognition Rate (Set 1)	Reject Rate (Set 1)	Recognition Rate (Set 2)	Reject Rate (Set 2)
20	99.84%	11.23%	99.49%	8.67%
24	99.98%	16.50%	99.90%	14.06%
28	99.99%	25.22%	99.99%	18.85%
32	100%	27.72%	99.99%	22.08%

Table 1. Recognition and reject rates on test sets with different threshold values

4.2. Verification

In on-line verification, a user indicates his/her identity. Therefore, the system matches the input fingerprint image only to his/her stored template. To determine the verification accuracy of our system, we used each one of our database images as template and all the others as input fingerprints which need to be verified. An input fingerprint is matched against the template. If the matching score is below the threshold value of 25, then the input fingerprint is rejected. If the matching score exceeds 25, then a valid verification is established. With this scheme, based on 89,700 (300 × 299) verifications, a 99.99% verification rate can be achieved with a tolerable reject rate (15%).

In order for an on-line fingerprint verification system to be acceptable in practice, the response time of the system needs to be within a few seconds. Table 2 shows that our online fingerprint verification system does meet the response time requirement of on-line verification.

Minutia Extraction (second)	Minutia Matching (second)	Total (second)
5.35	2.55	7.90

Table 2. Average CPU time for minutia extraction and matching on a SPARC 20 workstation.

5. Conclusions

We have introduced an on-line fingerprint verification system. The implemented minutia extraction algorithm is accurate and fast in minutia extraction. The alignment-based elastic matching algorithm is capable of finding the correspondences between minutiae without resorting to exhaustive search. It provides a good performance, because it has the ability to adaptively compensate for the nonlinear deformations and inexact pose transformations between different fingerprints. Experimental results show that our system achieves excellent performance in a real on-line verification environment. It meets the response time requirements of on-line verification with high accuracy.

Based on the experimental results, we observe that the matching errors of our system mainly result from (i) incorrect minutiae and (ii) inaccurate alignment. A number of factors are detrimental to the correct location of minutia. Among them, poor image quality is the most serious one. We are currently investigating a number of image enhancement schemes.

References

[1] N. Ansari, M. H. Chen, and E. S. H. Hou, A Genetic Algorithm for Point Pattern Matching, Chapter 13 in *Dynamic, Genetic, and Chaotic Programming* by B. Souĉek and the IRIS Group. John Wiley & Sons, 1992.

[2] Federal Bureau of Investigation, The Science of Fingerprints: Classification and Uses, U.S. Government Printing Office, Washington, D. C. 1984.

[3] S. Gold and A. Rangarajan, A Graduated Assignment Algorithm for Graph Matching, *IEEE Transactions on PAMI*, Vol. 18, No. 4, pp. 377-388, 1996.

[4] Henry C. Lee and R. E. Gaensslen, editors, Advances in Fingerprint Technology, Elsevier, New York, 1991.

[5] B. Miller, Vital Signs of Identity, *IEEE Spectrum*, Vol. 31, No. 2, pp. 22-30, 1994.

[6] A. Ranade and A Rosenfeld, Point Pattern Matching by Relaxation, *Pattern Recognition*, Vol. 12, No. 2, pp. 269-275, 1993.

[7] A. Ravishankar Rao, A Taxonomy for Texture Description and Identification, Springer-Verlag, New York, 1990.

[8] N. Ratha, S. Chen, and A. K. Jain, Adaptive Flow Orientation Based Feature Extraction in Fingerprint Images, *Pattern Recognition*, Vol. 28, No. 11, pp. 1657-1672, 1995.

[9] J. Ton and A. K. Jain, Registering Landsat Images by Point Matching, *IEEE Transactions on Geoscience and Remote Sensing*, Vol. 27, No. 5, pp. 642-651, 1989.

On-line Fingerprint Verification

Anil Jain and Lin Hong
Pattern Recognition and Image Processing Laboratory
Department of Computer Science
Michigan State University
East Lansing, MI 48824
{jain,honglin}@cps.msu.edu

Ruud Bolle
Exploratory Computer Vision Group
IBM T. J. Watson Research Center
Yorktown Heights, NY 10598
bolle@watson.ibm.com

November 26, 1996

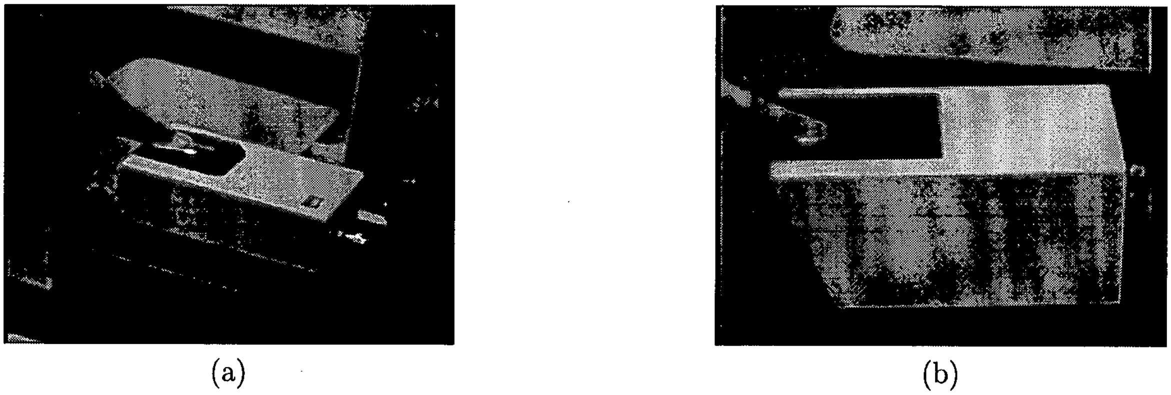
Abstract

Fingerprint verification is one of the most reliable personal identification methods. However, manual fingerprint verification is so tedious, time-consuming, and expensive that it is incapable of meeting today's increasing performance requirements. An automatic fingerprint identification system (AFIS) is widely needed. It plays a very important role in forensic and civilian applications such as criminal identification, access control, and ATM card verification. This paper describes the design and implementation of an on-line fingerprint verification system which operates in two stages: (i) minutia extraction and (ii) minutia matching. An improved version of the minutia extraction algorithm proposed by Ratha et al.*, which is much faster and more reliable, is implemented for extracting features from an input fingerprint image captured with an on-line* *inkless scanner. For minutia matching, an alignment-based elastic matching algorithm has been developed. This algorithm is capable of finding the correspondences between minutiae in the input image and the stored template without resorting to exhaustive search and has the ability of adaptively compensating for the nonlinear deformations and inexact pose transformations between fingerprints. The system has been tested on two sets of fingerprint images captured with inkless scanners. The verification accuracy is found to be acceptable. Typically, a complete fingerprint verification procedure takes, on an average, about 8 seconds on a SPARC 20 workstation. These experimental results show that our system meets the response time requirements of on-line verification with high accuracy.*

Keywords: biometrics, fingerprints, matching, verification, minutia, orientation field, ridge extraction.

1 Introduction

Fingerprints are graphical flow-like ridges present on human fingers. They have been widely used in personal identification for several centuries [11]. The validity of their use has been well established. Inherently, using current technology fingerprint identification is much more reliable than other kinds of popular personal identification methods based on signature, face, and speech [11, 3, 15]. Although fingerprint verification is usually associated with criminal identification and police work, it has now become more popular in civilian applications such as access control, financial security and verification of firearm purchasers and driver license applicants [11, 3]. Usually, fingerprint verification is performed manually by professional fingerprint experts. However, manual fingerprint verification is so tedious, time-consuming, and expensive that it does not meet the performance requirements of the new applications. As a result, automatic fingerprint identification systems (AFIS) are in great demand [11]. Although significant progress has been made in designing automatic fingerprint identification systems over the past thirty years, a number of design factors (lack of reliable minutia extraction algorithms, difficulty in quantitatively defining a reliable match between fingerprint images, fingerprint classification, *etc.*) create bottlenecks in achieving the desired performance [11].



(a) (b)

Figure 1: Inkless fingerprint scanners: (a) manufactured by *Identix*; (b) manufactured by *Digital Biometrics.*

An automatic fingerprint identification system is concerned with some or all of the following issues:

- Fingerprint acquisition. How to acquire fingerprint images and how to represent them in a proper format?
- Fingerprint verification. To determine whether two fingerprints are from the same finger.
- Fingerprint identification. To search for a query fingerprint in a database.

- Fingerprint classification. To assign a given fingerprint to one of the pre-specified categories according to its geometric appearance.

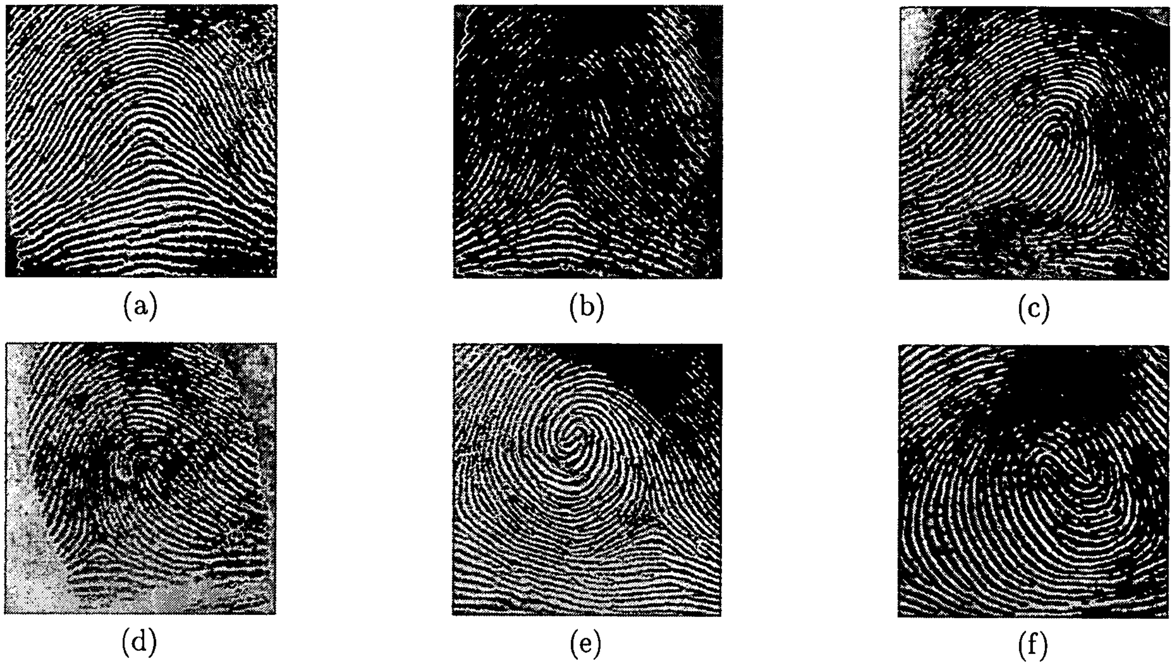
A number of methods are used to acquire fingerprints. Among them, the inked impression method remains the most popular. It has been essentially a standard technique for fingerprint acquisition for more than a hundred years [3]. The first step in capturing an inked impression of a fingerprint is to place a few dabs of ink on a slab and rolling it out smoothly with a roller until the slab is covered with a thin, even layer of ink. Then the finger is rolled from one side of the nail to the other side over the inked slab which inks the ridge patterns on top of the finger completely. After that, the finger is rolled on a piece of paper so that the inked impression of the ridge pattern of the finger appears on the paper. Obviously, this method is time-consuming and unsuitable for an on-line fingerprint verification system. Inkless fingerprint scanners are now available which are capable of directly acquiring fingerprints in digital form. This method eliminates the intermediate digitization process of inked fingerprint impressions and makes it possible to build an on-line system. Figure 1 shows the two inkless fingerprint scanners used in our verification system. Fingerprint images captured with the inked impression method and the inkless impression method are shown in Figure 2.

The goal of fingerprint classification is to assign a given fingerprint to a specific category according to its geometric properties (Figure 3 shows a coarse-level fingerprint classification). The main purpose of fingerprint classification is to facilitate the management of large fingerprint databases and to speedup the process of fingerprint matching. Generally, manual fingerprint classification is performed within a specific framework such as the well-known Henry system [3]. Different frameworks use different sets of properties. However, no matter

Figure 2: Comparison of fingerprint images captured with (a) inked impression method (from NIST database) and (b) inkless impression method (with a scanner manufactured by Digital Biometrics).

what type of framework is used, the classification is based on ridge patterns, local ridge orientations and minutiae. Therefore, if these properties can be described quantitatively and extracted automatically from a fingerprint image then fingerprint classification will become an easier task. During the past several years, a number of researchers have attempted to solve the fingerprint classification problem [11, 3, 9, 10, 26]. Unfortunately, their efforts have not resulted in the desired accuracy. Algorithms reported in the literature classify fingerprints into 5 or 6 categories with about 90% classification accuracy on a medium size test set (several thousand images) [9, 10, 26]. However, to achieve a higher recognition accuracy with a large number of categories still remains a difficult problem.

Fingerprint verification determines whether two fingerprints are from the same finger or not. It is widely believed that if two fingerprints are from the same source, then their local ridge structures (minutia details) match each other topologically [11, 3]. Eighteen different types of local ridge descriptions have been identified [11]. The two most prominent structures are ridge endings and ridge bifurcations which are usually called minutiae. Figure 4


(a) (b) (c)

(d) (e) (f)

Figure 3: A coarse-level fingerprint classification of six categories: (a) arch, (b) tented arch, (c) right loop, (d) left loop, (e) whorl, and (f) twin loop.

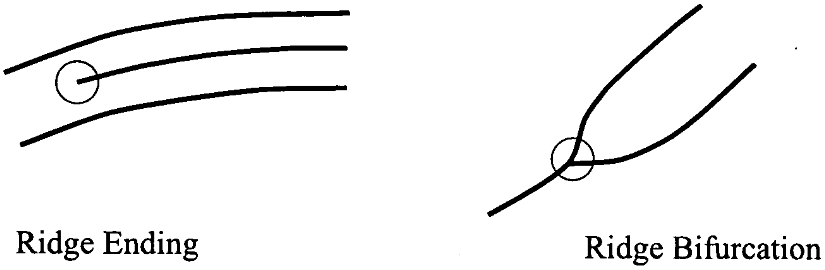


Figure 4: Ridge ending and ridge bifurcation.

shows examples of ridge endings and ridge bifurcations. Based on this observation and by representing the minutiae as a point pattern, an automatic fingerprint verification problem may be reduced to a point pattern matching (minutia matching) problem. In the ideal case, if (i) the correspondences between the template and input fingerprint are known, (ii) there are no deformations such as translation, rotation and nonlinear deformations, *etc.* between them, and (iii) each minutia present in a fingerprint image is exactly localized, then fingerprint verification consists of the trivial task of counting the number of spatially

Figure 5: Two different fingerprint images from the same finger. In order to know the correspondence between the minutiae of these two fingerprint images, all the minutiae must be precisely localized and the deformations must be recovered.

matching pairs between the two images. However, in practice (i) no correspondence is known beforehand, (ii) there are relative translation, rotation and nonlinear deformations between template minutiae and input minutiae, (iii) spurious minutiae are present in both templates and inputs, and (iv) some minutiae are missed. Therefore, in order for a fingerprint verification algorithm to operate under such circumstances, it is necessary to automatically obtain minutia correspondences, to recover deformations, and to detect spurious minutiae from fingerprint images. Unfortunately, this goal is quite difficult to achieve. Figure 5 illustrates the difficulty with an example of two fingerprint images of the same finger.

Fingerprint identification refers to the process of matching a query fingerprint against a given fingerprint database to establish the identity of an individual. Its goal is to quickly determine whether a query fingerprint is present in the database and to retrieve those which are most similar to the query from the database. The critical issues here are both retrieval speed and accuracy. In fact, this problem relates to a number of techniques studied under the auspices of computer vision, pattern recognition, database, and parallel processing. Operational fingerprint retrieval systems are being used by various law enforcement agencies [11].

In this paper, we will introduce an on-line fingerprint verification system whose purpose is to capture fingerprint images using an inkless scanner and to compare them with those stored in the database in "real time". Such a system has great utility in a variety of personal identification and access control applications. The overall block diagram of our system is shown in Figure 6. It operates as follows: (i) off-line phase: several impressions (depending on the specification of the system) of the fingerprint of a person to be verified are first captured and processed by a feature extraction module; the extracted features are stored as templates in a database for later use; (ii) on-line phase: the individual to be verified gives his/her identity and places his/her finger on the inkless fingerprint scanner, minutia points are extracted from the captured fingerprint image; these minutiae are then fed to a matching module, which matches them against his/her own templates in the database.

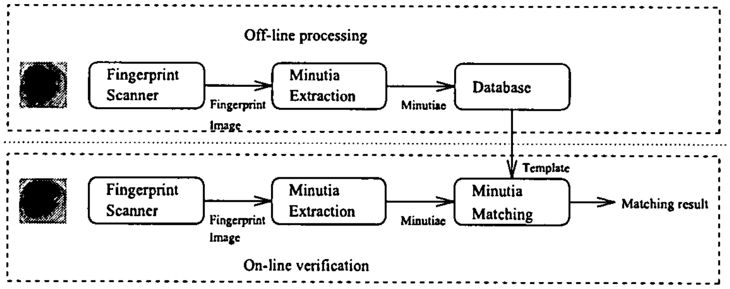


Figure 6: Overview of our on-line fingerprint verification system.

The following two modules are the main components of our on-line fingerprint verification system:

- *Minutiae extraction.* Minutiae are ridge endings or ridge bifurcations. Generally, if a perfect segmentation can be obtained, then minutia extraction is just a trivial task of extracting singular points in a thinned ridge map. However, in practice, it is not always possible to obtain a perfect ridge map. Some global heuristics need to be used to overcome this limitation.

- *Minutia matching.* Minutia matching, because of deformations in sensed fingerprints, is an elastic matching of point patterns without knowing their correspondences beforehand. Generally, finding the best match between two point patterns is intractable even if minutiae are exactly located and no deformations exist between these two point patterns. The existence of deformations makes the minutia matching much more difficult.

For segmentation and minutia extraction, a modified version of the minutia extraction algorithm proposed in [18] is implemented which is much faster and more reliable for minutia extraction. We propose a hierarchical approach to obtain a smooth orientation field estimate of the input fingerprint image, which greatly improves the performance of minutia extraction. For minutia matching, we propose an alignment-based elastic matching algorithm. This algorithm is capable of finding the correspondences between minutiae without resorting to an exhaustive search and has the ability to adaptively compensate for the nonlinear deformations and inexact pose transformations between different fingerprints. Experimental results show that our system achieves excellent performance in a real environment.

In the following sections we will describe in detail our on-line fingerprint verification system. Section 2 mainly discusses the fingerprint feature extraction module. Section 3 presents our minutia matching algorithm. Experimental results on two fingerprint databases captured with two different inkless scanners are described in section 4. Section 5 contains the summary and discussion.

2 Minutia Extraction

It is widely known that a professional fingerprint examiner relies on minute details of ridge structures to make fingerprint identifications [11, 3]. The topological structure of the minutiae of a fingerprint is unique and invariant with aging and impression deformations [11, 3]. This implies that fingerprint identification can be based on the topological structural matching of these minutiae. This reduces the complex fingerprint verification to minutia matching process which, in fact, is a sort of point pattern matching with the capability of tolerating, to some restricted extent, deformations of the input point patterns. Therefore, the first stage in an automatic fingerprint verification procedure is to extract minutiae from fingerprints. In our on-line fingerprint verification system, we have implemented a minutia extraction algorithm which is an improved version of the method proposed by Ratha *et al.* [18]. Its overall flowchart is depicted in Figure 7. We assume that the resolution of input fingerprint images is 500 dpi.

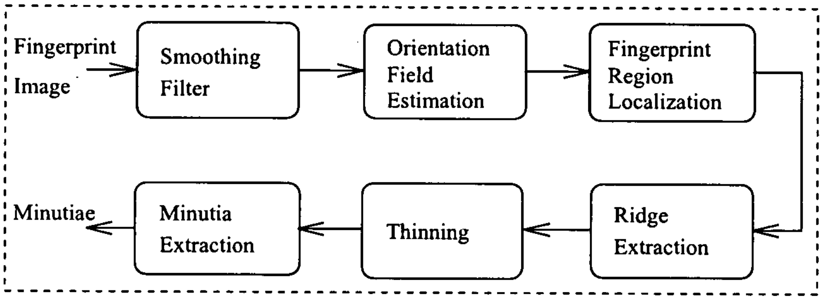


Figure 7: Flowchart of the minutia extraction algorithm.

2.1 Estimation of Orientation Field

A number of methods have been proposed to estimate the orientation field of flow-like patterns [17]. In our system, a new hierarchical implementation of Rao's algorithm [17] is used.

Rao's algorithm consists of the following main steps:

1. Divide the input fingerprint image into blocks of size $W \times W$.

2. Compute the gradients G_x and G_y at each pixel in each block.

3. Estimate the local orientation of each block using the following formula:

$$\theta_o = \frac{1}{2} tan^{-1}\left(\frac{\sum_{i=1}^{W}\sum_{j=1}^{W} 2G_x(i,j)G_y(i,j)}{\sum_{i=1}^{W}\sum_{j=1}^{W}(G_x^2(i,j) - G_y^2(i,j))}\right), \tag{1}$$

   where W is the size of the block, and G_x and G_y are the gradient magnitudes in x and y directions, respectively.

The orientation field of a good quality fingerprint image can be reasonably estimated with this algorithm. However, the presence of high-curvature ridges, noise, smudges, and breaks in ridges leads to a poor estimate of the local orientation field. A post-processing procedure needs to be applied to overcome this limitation. In our system, the following iterative steps are added to improve an inconsistent orientation field:

- Compute the *consistency level* of the orientation field in the local neighborhood of a block (i, j) with the following formula:

$$C_o = \frac{1}{N}\sqrt{\sum_{(i',j') \in D} |\theta(i',j') - \theta(i,j)|^2}, \tag{2}$$

$$|\theta' - \theta| = \begin{cases} d & \text{if } (d = (\theta' - \theta + 360) \bmod 360) < 180, \\ d - 180 & \text{otherwise,} \end{cases} \tag{3}$$

  where D represents the local neighborhood around the block (i, j) (in our system, the size of D is 5×5); N is the number of blocks within D; $\theta(i', j')$ and $\theta(i, j)$ are local ridge orientations at blocks (i', j') and (i, j), respectively.

- If the *consistency level* (Eq.(2)) is above a certain threshold T_c, then the local orientations around this region are re-estimated at a lower resolution level until it is below a certain level. With this post-smoothing scheme, a fairly smooth orientation field estimate can be obtained. Figure 8 shows the orientation field of a fingerprint image estimated with our new algorithm.

(a) Rao's method

(b) Hierarchical method

Figure 8: Comparison of orientation fields by Rao's method and the proposed hierarchical method; the block size ($W \times W$) is 16×16 and the size of D is 5×5.

After the orientation field of an input fingerprint image is estimated, a segmentation algorithm which is based on the local variance of grey level is used to locate the region of interest from the fingerprint image. In our segmentation algorithm, we assume that there is only one fingerprint present in the image.

2.2 Ridge Detection

After the orientation field of the input image is estimated and the fingerprint region is located, the next step of our minutia exaction algorithm is ridge detection. The most salient property corresponding to ridges in a fingerprint image is the fact that grey level values on ridges attain their local maxima along the normal directions of local ridges. Therefore, pixels can be identified to be ridge pixels based on this property. In our minutia detection algorithm, a fingerprint image is first convolved with the following two masks, $h_t(x,y;u,v)$ and $h_b(x,y;u,v)$, of size $L \times H$ (11×7 in our system), respectively. These two masks are capable of adaptively accentuating the local maximum grey level values along the normal direction of the local ridge direction:

$$
\begin{aligned}
h_t(x,y;u,v) &= \begin{cases} -\frac{1}{\sqrt{2\pi}\delta}e^{-\frac{u}{\delta^2}}, & \text{if } u = (v\tan(\theta(x,y)) - \frac{H}{2\cos(\theta(x,y))}), v \in \Omega \\ \frac{1}{\sqrt{2\pi}\delta}e^{-\frac{u}{\delta^2}}, & \text{if } u = (v\tan(\theta(x,y))), v \in \Omega \\ 0, & \text{otherwise,} \end{cases} && (4) \\
h_b(x,y;u,v) &= \begin{cases} -\frac{1}{\sqrt{2\pi}\delta}e^{-\frac{u}{\delta^2}}, & \text{if } u = (v\tan(\theta(x,y)) + \frac{H}{2\cos(\theta(x,y))}), v \in \Omega \\ \frac{1}{\sqrt{2\pi}\delta}e^{-\frac{u}{\delta^2}}, & \text{if } u = (v\tan(\theta(x,y))), v \in \Omega \\ 0, & \text{otherwise,} \end{cases} && (5) \\
\Omega &= \left[-\left|\frac{L\sin(\theta(x,y))}{2}\right|, \left|\frac{L\sin(\theta(x,y))}{2}\right| \right], && (6)
\end{aligned}
$$

where $\theta(x,y)$ represents the local ridge direction at pixel (x,y). If *both* the grey level values at pixel (x,y) of the convolved images are larger than a certain threshold T_{ridge}, then pixel (x,y) is labeled as a ridge. By adapting the mask width to the width of the local ridge, this algorithm can efficiently locate the ridges in a fingerprint image.

However, due to the presence of noise, breaks, and smudges, *etc.* in the input image, the resulting binary ridge map often contains holes and speckles. When ridge skeletons are used for the detection of minutiae, the presence of such holes and speckles will severely handicap the performance of our minutia extraction algorithm because these holes and speckles may drastically change the skeleton of the ridges. Therefore, a hole and speckle removal procedure needs to be applied before ridge thinning.

After the above steps are performed on an input fingerprint image, a relatively smooth ridge map of the fingerprint is obtained. The next step of our minutia detection algorithm is to thin the ridge map and locate the minutiae.

2.3 Minutia Detection

Minutia detection is a trivial task when an ideal thinned ridge map is obtained. Without a loss of generality, we assume that if a pixel is on a thinned ridge (8-connected), then it has a value 1, and 0 otherwise. Let (x, y) denote a pixel on a thinned ridge, and $N_0, N_1, ..., N_7$ denote its 8 neighbors. A pixel (x, y) is a ridge ending if $(\sum_{i=0}^{8} N_i) = 1$ and a ridge bifurcation if $(\sum_{i=0}^{8} N_i) > 2$. However, the presence of undesired spikes and breaks present in a thinned ridge map may lead to many spurious minutiae being detected. Therefore, before the minutia detection, a smoothing procedure is applied to remove spikes and to join broken ridges. Our ridge smoothing algorithm uses the following heuristics:

- If a branch in a ridge map is roughly orthogonal to the local ridge directions and its length is less than a specified threshold T_b, then it will be removed.
- If a break in a ridge is short enough and no other ridges pass through it, then it will be connected.

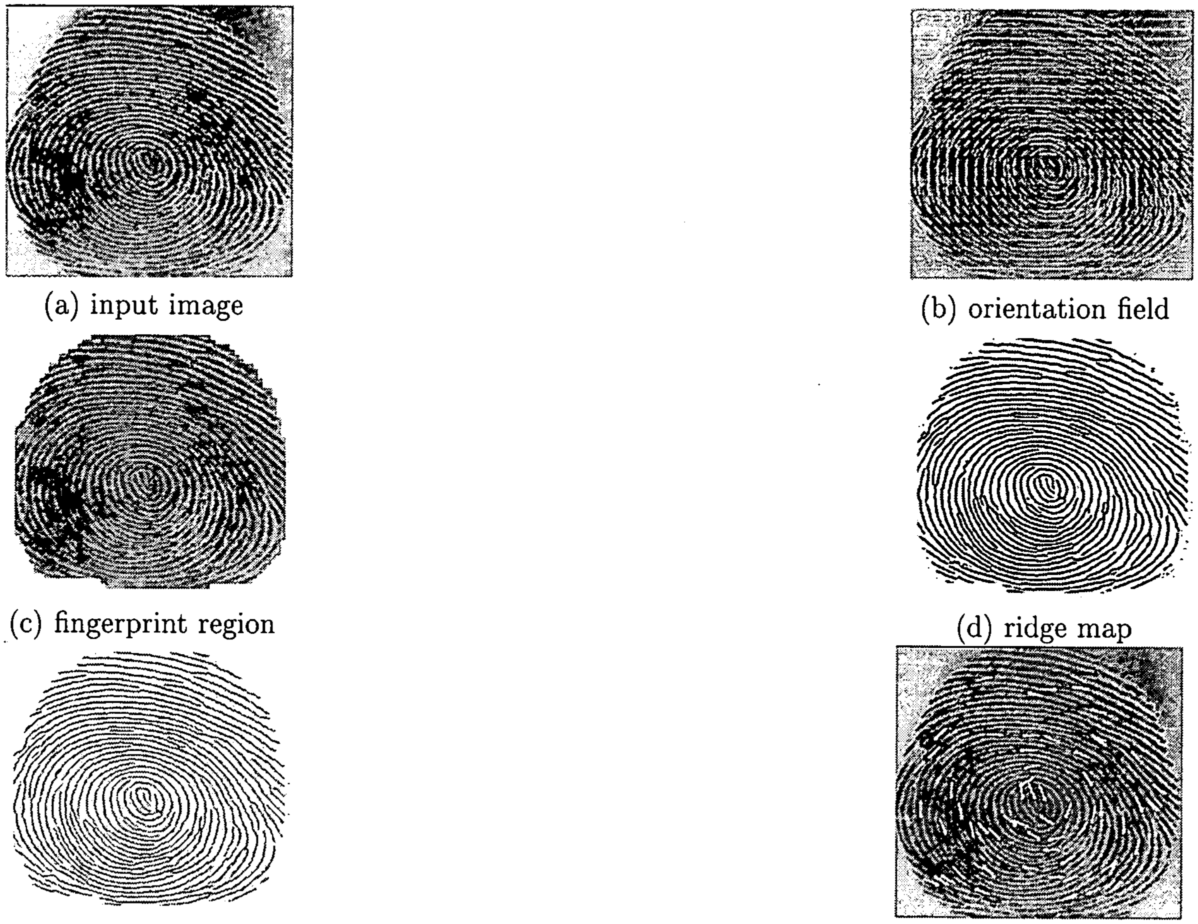
Although the above heuristics do delete a large percentage of spurious minutiae, many spurious minutiae still survive. The reason is that the above processing relies on local ridge information. If this information itself is unreliable, then the above heuristics have no way of differentiating false minutiae from true minutiae. Therefore, a refinement which is based on structural information is necessary. Our refinement algorithm eliminates the spurious minutiae based on the following rules:

- If several minutiae form a cluster in a small region, then remove all of them except for the one nearest to the cluster center.
- If two minutiae are located close enough, facing each other, but no ridges lie between them, then remove both of them.

After the above refinement procedure is performed, the surviving minutiae are treated as true minutiae. Although the above heuristics can not ensure a perfect location of each minutia, they are able to delete several spurious minutiae. For each surviving minutia, the following parameters are recorded: (i) x-coordinate, (ii) y-coordinate, (iii) orientation which is defined as the local ridge orientation of the associated ridge, and (iv) the associated ridge. The recorded ridges are represented as one-dimensional discrete signals which are normalized by the average inter-ridge distance. These recorded ridges are used for alignment in the minutia matching phase. Figure 9 shows the results of our minutia extraction algorithm on a fingerprint image captured with an inkless scanner.

3 Minutia Matching

Generally, an automatic fingerprint verification/identification is achieved with point pattern matching (minutiae matching) instead of a pixel-wise matching or a ridge pattern matching of fingerprint images. A number of point pattern matching algorithms have been proposed in the literature [23, 1, 21, 16]. Because a general point matching problem is essentially



(a) input image

(b) orientation field

(c) fingerprint region

(d) ridge map

(e) thinned ridge map

(f) extracted minutiae

Figure 9: Results of our minutia extraction algorithm on a fingerprint image (512 × 512) captured with an inkless scanner; (a) input image; (b) orientation field superimposed on the input image; (c) fingerprint region; (d) extracted ridges; (e) thinned ridge map; (f) extracted minutiae and their orientations superimposed on the input image.

intractable, features associated with each point and their spatial properties such as the relative distances between points are often used in these algorithms to reduce the exponential number of search paths.

The relaxation approach [16] iteratively adjusts the confidence level of each corresponding pair based on its consistency with other pairs until a certain criterion is satisfied. Although a number of modified versions of this algorithm have been proposed to reduce the matching complexity [23], these algorithms are inherently slow because of their iterative nature.

The Hough transform-based approach proposed by Stockman *et al.* [22] converts point pattern matching to a problem of detecting the highest peak in the Hough space of transformation parameters. It discretizes the transformation parameter space and accumulates evidence in the discretized space by deriving transformation parameters that relate two point patterns using a substructure or feature matching technique. Karu and Jain [8] proposed a hierarchical Hough transform-based registration algorithm which greatly reduced the size of accumulator array by a multi-resolution approach. However, if the number of minutia point is less than 30, then it is very difficult to accumulate enough evidence in the Hough transform space for a reliable match.

Another approach to point matching is based on energy minimization. This approach defines a cost function based on an initial set of possible correspondences and uses an appropriate optimization algorithm such as genetic algorithm [1] and simulated annealing [21] to find a possible suboptimal match. These methods tend to be very slow and are unsuitable for an on-line fingerprint verification system.

In our system, an alignment-based matching algorithm is implemented. Recognition by alignment has received a great deal of attention during the past few years [12], because it is simple in theory, efficient in discrimination, and fast in speed. Our alignment-based matching algorithm decomposes the minutia matching into two stages: (i) *Alignment stage*, where transformations such as translation, rotation and scaling between an input and a template in the database are estimated and the input minutiae are aligned with the template minutiae according to the estimated parameters; and (ii) *Matching stage*, where both the input minutiae and the template minutiae are converted to polygons in the polar coordinate system and an elastic string matching algorithm is used to match the resulting polygons.

3.1 Alignment of Point Patterns

Ideally, two sets of planar point patterns can be aligned completely by two corresponding point pairs. A true alignment between two point patterns can be obtained by testing all possible corresponding point pairs and selecting the optimal one. However, due to the presence of noise and deformations, the input minutiae cannot always be aligned exactly with respect to those of the templates. In order to accurately recover pose transformations between two point patterns, a relatively large number of corresponding point pairs need to be used. This leads to a prohibitively large number of possible correspondences to be tested. Therefore, an alignment by corresponding point pairs is not practical even though it is feasible.

It is well known that corresponding curve segments are capable of aligning two point patterns with a high accuracy in the presence of noise and deformations. Each minutia in a fingerprint is associated with a ridge. It is clear that a true alignment can be achieved by aligning corresponding ridges (see Figure 10). During the minutiae detection stage,

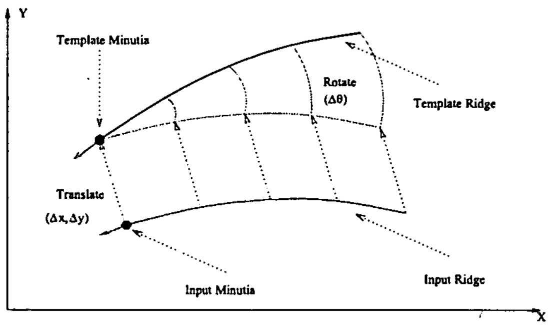


Figure 10: Alignment of the input ridge and the template ridge.

when a minutia is extracted and recorded, the ridge on which it resides is also recorded. This ridge is represented as a planar curve with its origin coincident with the minutia and its x-coordinate being in the same direction as the direction of the minutia. Also, this planar curve is normalized with the average inter-ridge distance. By matching these ridges, the relative pose transformation between the input fingerprint and the template can be accurately estimated. To be specific, let R^d and R^D denote the sets of ridges associated with the minutiae in input image and template, respectively. Our alignment algorithm can be described in terms of the following steps:

1. For each ridge $d \in R^d$, represent it as an one-dimensional discrete signal and match it against each ridge, $D \in R^D$ according to the following formula:

$$S = \frac{\sum_{i=0}^{L} d_i D_i}{\sqrt{\sum_{i=0}^{L} d_i^2 D_i^2}}, \tag{7}$$

   where L is the minimal length of the two ridges and d_i and D_i represent the distances from point i on the ridges d and D to the x-axis, respectively. The sampling interval on a ridge is set to the average inter-ridge distance. If the matching score S $(0 \leq S \leq 1)$ is larger than a certain threshold T_r, then go to step 2, otherwise continue to match the next pair of ridges.

2. Estimate the pose transformation between the two ridges (Figure 10). Generally, a least-square method can be used to estimate the pose transformation. However, in our system, we observe that the following method is capable of achieving the same accuracy with less computation. The translation vector $(\Delta x, \Delta y)^T$ between the two corresponding ridges is computed by

$$\begin{pmatrix} \Delta x \\ \Delta y \end{pmatrix} = \begin{pmatrix} x^d \\ y^d \end{pmatrix} - \begin{pmatrix} x^D \\ y^D \end{pmatrix}, \tag{8}$$

where $(x^d, y^d)^T$ and $(x^D, y^D)^T$ are the x and y coordinates of the two minutiae, which are called reference minutiae, associated with the ridges d and D, respectively. The rotation angle $\Delta\theta$ between the two ridges is computed by

$$\Delta\theta = \frac{1}{L}\sum_{i=0}^{L}(\gamma_i - \Gamma_i), \tag{9}$$

where L is the minimal length of the two ridges d and D; γ_i and Γ_i are radial angles of the ith point on the ridge with respect to the reference minutia associated with the two ridges d and D, respectively. The scaling factor between the input and template images is assumed to be 1. This is reasonable, because fingerprint images are captured with the same device in both the off-line processing phase and the on-line verification phase.

3. Denote the minutia $(x^d, y^d, \theta^d)^T$, based on which the pose transformation parameters are estimated, as the reference minutia. Translate and rotate all the N input minutiae

with respect to this reference minutia, according to the following formula:

$$\begin{pmatrix} x_i^A \\ y_i^A \\ \theta_i^A \end{pmatrix} = \begin{pmatrix} \Delta x \\ \Delta y \\ \Delta\theta \end{pmatrix} + \begin{pmatrix} \cos\Delta\theta & \sin\Delta\theta & 0 \\ \sin\Delta\theta & -\cos\Delta\theta & 0 \\ 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} x_i - x^d \\ y_i - y^d \\ \theta_i - \theta^d \end{pmatrix}, \qquad (10)$$

where $(x_i, y_i, \theta_i)^T$, $(i = 1, 2, ..., N)$, represents an input minutia and $(x_i^A, y_i^A, \theta_i^A)^T$ represents the corresponding aligned minutia.

3.2 Aligned Point Pattern Matching

If two identical point patterns are exactly aligned with each other, each pair of corresponding points are completely coincident. In such a case, a point pattern matching can be simply achieved by counting the number of overlapping pairs. However, in practice, such a situation is not encountered. On the one hand, the error in determining and localizing minutia hinders the alignment algorithm to recover the relative pose transformation exactly, while on the other hand, our alignment scheme described above does not model the nonlinear deformation of fingerprints which is an inherent property of fingerprint impressions. With the existence of such a nonlinear deformation, it is impossible to exactly recover the position of each input minutia with respect to its corresponding minutia in the template. Therefore, the aligned point pattern matching algorithm needs to be elastic which means that it should be capable of tolerating, to some extent, the deformations due to inexact extraction of minutia positions and nonlinear deformations. Usually, such an elastic matching can be achieved by placing a bounding box around each template minutia, which specifies all the possible positions of the corresponding input minutia with respect to the template minutia, and restricting the corresponding minutia in the input image to be within this box [18]. This method does not provide a satisfactory performance in practice, because local deformations may be small while the accumulated global deformations can be quite large. We have implemented an adaptive elastic matching algorithm with the ability to compensate the minutia localization errors and nonlinear deformations.

Let $P = ((x_1^P, y_1^P, \theta_1^P)^T, ..., (x_M^P, y_M^P, \theta_M^P)^T)$ denote the set of M minutiae in the template and $Q = ((x_1^Q, y_1^Q, \theta_1^Q)^T, ..., (x_N^Q, y_N^Q, \theta_N^Q)^T)$ denote the set of N minutiae in the input image which is aligned with the above template with respect to a given reference minutia point. The steps in our elastic point pattern matching algorithm are given below:

1. Convert each minutia point to the polar coordinate system with respect to the corresponding reference minutia on which the alignment is performed:

$$\begin{pmatrix} r_i \\ e_i \\ \theta_i \end{pmatrix} = \begin{pmatrix} \sqrt{(x_i^* - x^r)^2 + (y_i^* - y^r)^2} \\ \tan^{-1}(\frac{y_i^* - y^r}{x_i^* - x^r}) \\ \theta_i^* - \theta^r \end{pmatrix}, \qquad (11)$$

   where $(x_i^*, y_i^*, \theta_i^*)^T$ are the coordinates of a minutia, $(x^r, y^r, \theta^r)^T$ are the coordinates of the reference minutia, and $(r_i, e_i, \theta_i)^T$ is the representation of the minutia in polar coordinate system (r_i represents the radial distance, e_i represents the radial angle and θ_i represents the orientation of the minutia with respect to the reference minutia).

2. Represent the template and the input minutiae in the polar coordinate system as symbolic strings by concatenating each minutia in the increasing order of radial angles:

$$P_p = ((r_1^P, e_1^P, \theta_1^P)^T, ..., (r_M^P, e_M^P, \theta_M^P)^T) \qquad (12)$$

$$Q_p = ((r_1^Q, e_1^Q, \theta_1^Q)^T, ..., (r_N^Q, e_N^Q, \theta_N^Q)^T), \qquad (13)$$

where (r_*^P, e_*^P, θ_*^P) and (r_*^Q, e_*^Q, θ_*^Q) represent the corresponding radius, radial angle, and normalized minutia orientation with respect to the reference minutia, respectively.

3. Match the resulting strings P_p and Q_p with a dynamic-programming algorithm [4] to find the edit distance between P_p and Q_p which is described below.

4. Use the edit distance between P_p and Q_p to establish the correspondence of the minutiae between P_p and Q_p. The matching score, M_{pq}, is then computed according to the following formula:

$$M_{pq} = \frac{100 N_{pair}}{\max\{M, N\}}, \tag{14}$$

where N_{pair} is the number of the minutiae which fall in the bounding boxes of template minutiae. The maximum and minimum values of the matching score are 100 and 1, respectively. The former value indicates a perfect match, while the later value indicates no match at all.

Minutia matching in the polar coordinate has several advantages. We have observed that the nonlinear deformation of fingerprints has a radial property. In other words, the nonlinear deformation in a fingerprint impression usually starts from a certain point (region) and nonlinearly radiates outward. Therefore, it is beneficial to model it in the polar space. At the same time, it is much easier to formulate rotation, which constitutes the main part of the alignment error between an input image and a template, in the polar space than in the Cartesian space. The symbolic string generated by concatenating points in an increasing order of radial angle in polar coordinate uniquely represents a point pattern. This reveals that the point pattern matching can be achieved with a string matching algorithm.

A number of string matching algorithms have been reported in the literature [4]. Here, we are interested in incorporating an elastic criteria into a string matching algorithm. Generally, string matching can be thought of as the maximization/minimization of a certain cost function such as the edit distance. Intuitively, including an elastic term in the cost function of a string matching algorithm can achieve a certain amount of error tolerance. Given two strings P_p and Q_p of lengths M and N, respectively, the edit distance, $C(M, N)$, in our algorithm is recursively defined with the following equations:

$$C(m,n) = \begin{cases} 0 & \text{if } m = 0 \text{ or } n = 0 \\ \min \left\{ \begin{array}{c} C(m-1,n) + \Omega \\ C(m,n-1) + \Omega \\ C(m-1,n-1) + w(m,n) \end{array} \right\} & 0 < m \leq M \text{ and } 0 < n \leq N, \end{cases} \tag{15}$$

$$w(m,n) = \begin{cases} \alpha \left| r_m^P - r_n^Q \right| + \beta \Delta e + \gamma \Delta \theta & \text{if } \left| r_m^P - r_n^Q \right| < \delta,\ \Delta e < \epsilon \text{ and } \Delta \theta < \varepsilon \\ \Omega & \text{otherwise,} \end{cases} \tag{16}$$

$$\Delta e = \begin{cases} a & \text{if } (a = (e_m^P - e_n^Q + 360) \bmod 360) < 180 \\ a - 180 & \text{otherwise,} \end{cases} \tag{17}$$

$$\Delta \theta = \begin{cases} a & \text{if } (a = (\theta_m^P - \theta_n^Q + 360) \bmod 360) < 180 \\ a - 180 & \text{otherwise,} \end{cases} \tag{18}$$

where α, β, and γ are the weights associated with each component, respectively; δ, ϵ and ε specify the bounding box; and Ω is a pre-specified penalty for a mismatch. Such an edit distance, to some extent, captures the elastic property of string matching. It represents a cost of changing one polygon to the other. However, this scheme can only tolerate, but not compensate for, the adverse effect on matching produced by the inexact localization of minutia and nonlinear deformations. Therefore, an adaptive mechanism is needed. This adaptive mechanism should be able to track the local nonlinear deformation and inexact alignment and try to alleviate them during the minimization process. However, we do not expect that this adaptive mechanism can handle the "order flip" of minutiae, which, to some extent, can be solved by an exhaustive re-ordering and matching within a local angular window.

In our matching algorithm, the adaptation is achieved by adjusting the bounding box (Figure 11) when an inexact match is found during the matching process. It can be represented as follows:

$$w'(m,n) = \begin{cases} \alpha\left|r^P_m - r^Q_n\right| + \beta\Delta e + \gamma\Delta\theta & \text{if} \begin{cases} \delta_l(m,n) < (r^P_m - r^Q_n) < \delta_h(m,n) \\ \epsilon_l(m,n)) < \Delta e < \epsilon_h(m,n) \\ \Delta\theta < \varepsilon \end{cases} \\ \Omega & \text{otherwise,} \end{cases} \tag{19}$$

$$\begin{pmatrix} \Delta r_a \\ \Delta e_a \end{pmatrix} = \begin{cases} \begin{pmatrix} r^P_m - r^Q_n \\ \Delta e \end{pmatrix} & \text{if} \begin{cases} \delta_l(m,n) < (r^P_m - r^Q_n) < \delta_h(m,n) \\ \epsilon_l(m,n)) < \Delta e < \epsilon_h(m,n) \\ \Delta\theta < \varepsilon \end{cases} \\ 0 & \text{otherwise,} \end{cases} \tag{20}$$

$$\delta_l(m+1,n+1) = \delta_l(m,n) + \eta\Delta r_a, \tag{21}$$

$$\delta_h(m+1,n+1) = \delta_h(m,n) + \eta\Delta r_a, \tag{22}$$

$$\epsilon_l(m+1,n+1) = \epsilon_l(m,n) + \eta\Delta e_a, \tag{23}$$

$$\epsilon_h(m+1,n+1) = \epsilon_h(m,n) + \eta\Delta e_a, \tag{24}$$

where $w'(m,n)$ represents the penalty for matching a pair of minutiae $(r^P_m, e^P_m, \theta^P_m)^T$ and

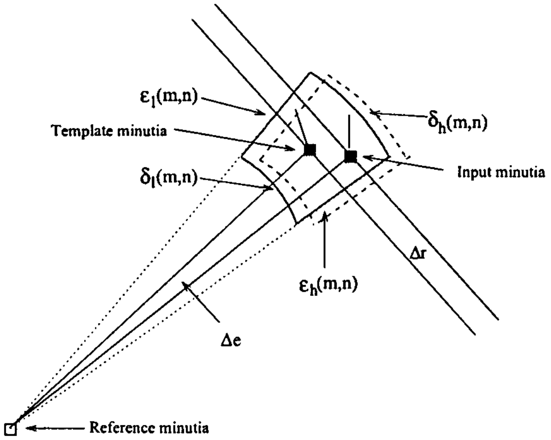


Figure 11: Bounding box and its adjustment.

$(r_n^Q, e_n^Q, \theta_n^Q)^T$, $\delta_l(m,n)$, $\delta_h(m,n)$, $\epsilon_l(m,n)$, and $\epsilon_h(m,n)$ specify the adaptive bounding box in the polar coordinate system (radius and radial angle); and η is the learning rate. This elastic string matching algorithm has a number of parameters which are critical to its performance. We have empirically determined the values of these parameters as follows: $\delta_l(0,0) = -8$; $\delta_h(0,0) = +8$; $\epsilon_l(0,0) = -7.5$; $\epsilon_h(0,0) = +7.5$; $\varepsilon = 30$; $\alpha = 1.0$; $\beta = 2.0$; $\gamma = 0.1$; $\Omega = 200(\alpha + \beta + \gamma)$; $\eta = 0.5$. The values of $\delta_l(0,0)$, $\delta_h(0,0)$, $\epsilon_l(0,0)$, and $\epsilon_h(0,0)$ depend on the resolution of fingerprint images. Figure 12 shows the results of applying the matching algorithm to an input minutia set and a template.

4 Experimental Results

We have tested our on-line fingerprint verification system on two sets of fingerprint images captured with two different inkless fingerprint scanners. Set 1 contains 10 images per finger from 18 individuals for a total of 180 fingerprint images, which were captured with a scanner manufactured by Identix. The size of these images is 380 × 380. Set 2 contains 10 images per

(a) (b) (c) (d)

Figure 12: Results of applying the matching algorithm to an input minutia set and a template; (a) input minutia set; (b) template minutia set; (c) alignment result based on the minutiae marked with green circles; (d) matching result where template minutiae and their correspondences are connected by green lines.

finger from 61 individuals for a total of 610 fingerprint images, which were captured with a scanner manufactured by Digital Biometrics. The size of these images is 640 × 480. When these fingerprint images were captured, no restrictions on the position and orientation of fingers were imposed. The captured fingerprint images vary in quality. Figures 13 and 14 show some of the fingerprint images in our database. Approximately 90% of the fingerprint images in our database are of reasonable quality similar to those shown in Figures 13 and 14, while about 10% of the fingerprint images in our database are not of good quality (Figure 15), which are mainly due to large creases and smudges in ridges and dryness of the impressed finger. First, we report some initial results on fingerprint matching, followed by fingerprint verification. The reasons why we did not use NIST-4 fingerprint database [25] to test the performance of our system are as follows: (*i*) we concentrate on live-scan verification, and (*ii*) NIST-4 fingerprint database is a very difficult fingerprint database which contains a large number of fingerprint images of poor quality and no result has been reported from other on-line verification systems for comparison.

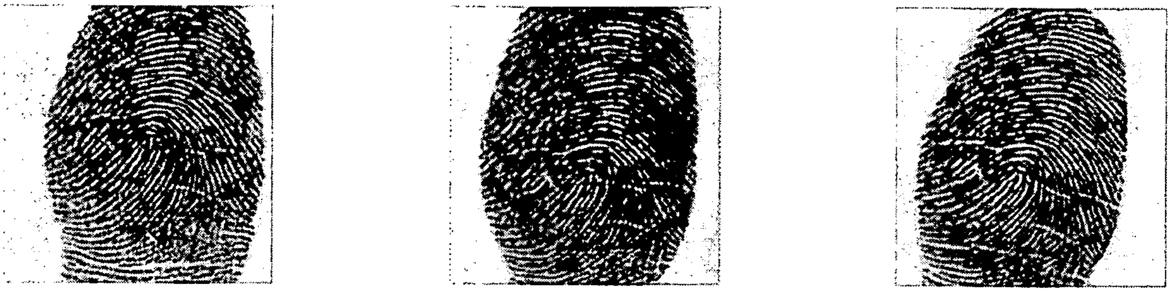


Figure 13: Fingerprint images captured with a scanner manufactured by Identix; the size of these images is 380 × 380; all the three images are from the same individual's finger.

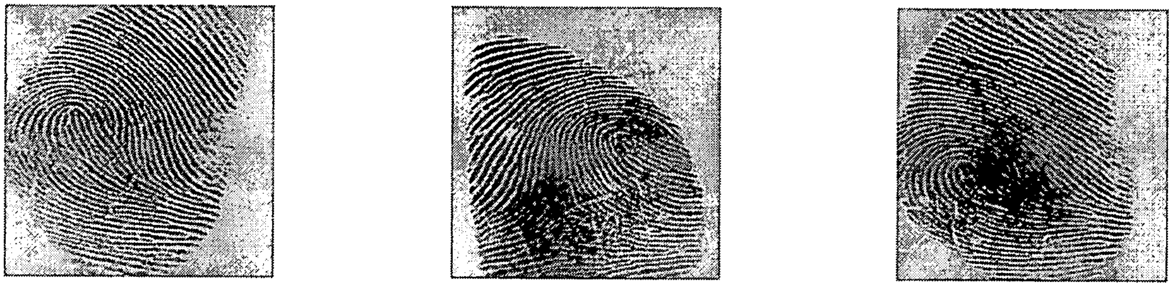
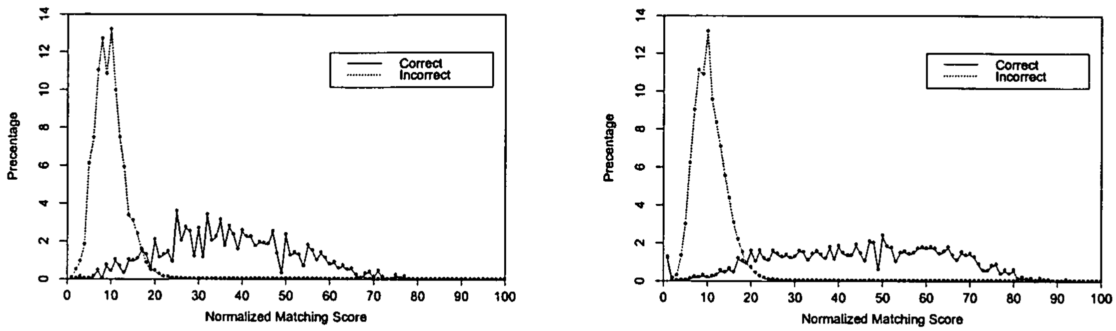


Figure 14: Fingerprint images captured with a scanner manufactured by Digital Biometrics; the size of these images is 640 × 480; all the three images are from the same individual's finger.

Figure 15: Fingerprint images of poor quality.



(a) Identix (b) Digital Biometrics

Figure 16: Distributions of correct and incorrect matching scores; vertical axis represents distribution of matching scores in percentage; (a) distribution of matching scores on test set 1 (180 images); (b) distribution of matching scores on test set 2 (610 images).

4.1 Matching

Each fingerprint in the test set was matched with the other fingerprints in the set. A matching was labeled correct if the matched fingerprint was among the 9 other fingerprints of the same individual, and incorrect otherwise. A total of 32,220 (180 × 179) matchings have been performed on test set 1 and 371,490 (610 × 609) matchings on test set 2. The distributions of correct and incorrect matching scores are shown in Figure 16. It can be seen from this figure that there exist two peaks in the distribution of matching scores. One pronounced peak corresponds to the incorrect matching scores which is located at a value around 10, and the other peak which resides at a value of 40 is associated with the correct matching scores. This indicates that our algorithm is capable of differentiating fingerprints at a high correct rate by setting an appropriate value of the threshold. Table 1 shows the verification rates and reject rates with different threshold values. The reject rate is defined as the percentage of correct fingerprints with their matching scores below the threshold value. As we have observed, both the incorrect matches and the high reject rates are due to fingerprint images with poor quality such as those shown in Figure 15. We can improve these matching results by ensuring that the database does not contain such poor quality fingerprint images.

Threshold	Verification	Reject
Value	Rate	Rate
20	99.839%	11.23%
22	99.947%	13.33%
24	99.984%	16.48%
26	99.994%	20.49%
28	99.996%	25.19%
30	100%	27.72%

(a)

Threshold	Verification	Reject
Value	Rate	Rate
20	99.426%	11.23%
22	99.863%	14.55%
24	99.899%	16.78%
26	99.969%	20.20%
28	99.989%	23.15%
30	99.999%	27.45%

(b)

Table 1: The verification rates and reject rates on test sets with different threshold values; (a) using Identix system (180 images); (b) using Digital Biometrics system (610 images).

4.2 Verification

In on-line verification, a user indicates his/her identity. Therefore, the system matches the input fingerprint image only to his/her stored templates. To determine the verification accuracy of our system, we used each one of our database images as an input fingerprint which needs to be verified. An input fingerprint image was matched against all the 9 other images of the same finger. If more than one half of the 9 matching scores exceeded the threshold value of 25, then the input fingerprint image is said to be from the same finger as the templates and a valid verification is established. With this scheme, a 100% verification rate can be achieved with a reject rate around 16% on both test sets. Again, this reject rate can be reduced by preprocessing the database to remove the stored templates of poor quality. This demonstrates that, in practice, using a k-nearest neighbor type of matching is adequate for a successful verification. Table 2 shows the matching rate which is defined as the percentage of the correct fingerprints (of the same finger) present among the best n ($n = 1, ...9$) matches.

For an on-line fingerprint verification system to be acceptable in practice, its response time needs to be within a few seconds. Table 3 shows the CPU requirements of our system. The CPU time for one verification, including fingerprint image acquisition, minutia extraction and minutia matching, is, on an average, approximately 8 seconds on a SPARC 20 workstation. It indicates that our on-line fingerprint verification system does meet the response time requirement of on-line verification.

The number of tests done on an automatic fingerprint identification system is never enough. Performance measures are as much a function of the algorithm as they are a function of the database used for testing. The biometrics community is slow at establishing benchmarks and the ultimate performance numbers of a fingerprint verification system are those which you find in a deployed system. Therefore, one can carry out only a limited amount of testing in a laboratory environment to show the anticipated system performance. Even in field testing, real performance numbers are not important - it's often the perceived performance which is crucial.

5 Conclusions

We have designed and implemented an on-line fingerprint verification system which operates in two stages: (i) minutia extraction, and (ii) minutia matching. A modified version of the minutia extraction algorithm proposed in [18] is used in our system which is much

Number of best matches	Matching Rate
9	91.17%
8	94.72%
7	96.89%
6	98.17%
5	98.89%
4	99.39%
3	99.72%
2	99.83%
1	99.94%

(a)

Number of best matches	Matching Rate
9	92.13%
8	94.40%
7	97.06%
6	97.67%
5	98.44%
4	99.11%
3	99.70%
2	99.79%
1	99.91%

(b)

Table 2: Matching rates on test sets using the leave-one-out method: (a) using Identix system (180 images); (b) using Digital Biometrics system (610 images).

Minutia Extraction (seconds)	Minutia Matching (seconds)	Total (seconds)
5.35	2.55	7.90

Table 3: Average CPU time for minutia extraction and matching on a SPARC 20 workstation.

faster and more reliable. A new hierarchical orientation field estimation algorithm results in a smoother orientation field which greatly improves the performance of the minutia extraction. An alignment-based elastic matching algorithm is proposed for minutia matching. This algorithm is quite fast, because it is capable of finding the correspondences between minutia points without resorting to an exhaustive search. At the same time, this matching algorithm has a good performance, because it has the ability to adaptively compensate for the nonlinear deformations and inexact pose transformations between different fingerprints. Experimental results show that our system achieves excellent performance in a realistic operating environment. It also meets the response time requirement of on-line verification.

Based on the experimental results, we observe that the matching errors in our system mainly result from (i) incorrect minutiae extraction, and (ii) inaccurate alignment. We observe that a number of factors are detrimental to the correct location of minutia. Among them, poor image quality is the most serious one. Therefore, in the future, our efforts will be focused on global image enhancement schemes. Another issue related to minutia detection is to incorporate a structural-based model in minutia detection which extracts minutiae based on their local ridge formations. For elastic matching, an important aspect is to utilize additional information (*e.g.*, neighboring ridges) about a minutia to increase the accuracy of alignment.

Acknowledgments

We gratefully acknowledge our useful discussions with Nalina Ratha and Shaoyun Chen and we would like to thank Nalina Ratha and Chitra Dorai for their careful comments on earlier versions of this paper.

References

[1] N. Ansari, M. H. Chen and E. S. H. Hou, A Genetic Algorithm for Point Pattern Matching, Chapter 13 in *Dynamic, Genetic, and Chaotic Programming* by B. Souĉek and the IRIS Group. John Wiley & Sons, 1992.

[2] P. E. Danielsson and Q. Z. Ye, Rotation-Invariant Operators Applied to Enhancement of Fingerprints, *Proc. 8th ICPR*, Rome, pp. 329-333, 1988.

[3] Federal Bureau of Investigation, The Science of Fingerprints: Classification and Uses, U.S. Government Printing Office, Washington, D. C., 1984.

[4] T. H. Cormen, C. E. Leiserson and R. L. Rivest, Introduction to Algorithms, McGraw-Hill, New York, 1990.

[5] D. C. Douglas Hung, Enhancement and Feature Purification of Fingerprint Images, *Pattern Recognition*, Vol. 26, No. 11, pp. 1661-1671, 1993.

[6] S. Gold and A. Rangarajan, A Graduated Assignment Algorithm for Graph Matching, Research Report YALEU/DCS/RR-1062, Yale University, Department of Computer Science, 1995.

[7] L. O'Gorman and J. V. Nickerson, An Approach to Fingerprint Filter Design, *Pattern Recognition*, Vol. 22, No. 1, pp. 29-38, 1989.

[8] K. Karu and A. K. Jain, Fingerprint Registration, *Research Report*, Michigan State University, Department of Computer Science, 1995.

[9] K. Karu and A. K. Jain, Fingerprint Classification, *Pattern Recognition,* Vol. 29, No. 3, pp. 389-404, 1996.

[10] M. Kawagoe and A. Tojo, Fingerprint Pattern Classification, *Pattern Recognition*, Vol. 17, No. 3, pp. 295-303, 1984.

[11] H. C. Lee and R. E. Gaensslen, editors, Advances in Fingerprint Technology, Elsevier, New York, 1991.

[12] D. P. Huttenlocher and S. Ullman, Object Recognition Using Alignment, *Proc. First Intern. Conf. Comput. Vision*, London, pp. 102-111, 1987.

[13] Z. R. Li and D. P. Zhang, A Fingerprint Recognition System With Micro-Computer, *Proc. 6th ICPR*, Montreal, pp. 939-941, 1984.

[14] L. Coetzee and E. C. Botha, Fingerprint Recognition in Low Quality Images, *Pattern Recognition*, Vol. 26, No. 10, pp. 1441-1460, 1993.

[15] B. Miller, Vital Signs of Identity, *IEEE Spectrum,* Vol. 31, No. 2, pp. 22-30, 1994.

[16] A. Ranade and A Rosenfeld, Point Pattern Matching by Relaxation, *Pattern Recognition*, Vol. 12, No. 2, pp. 269-275, 1993.

[17] A. Ravishankar Rao, A Taxonomy for Texture Description and Identification, Springer-Verlag, New York, 1990.

[18] N. Ratha, S. Chen and A. K. Jain, Adaptive Flow Orientation Based Feature Extraction in Fingerprint Images, *Pattern Recognition*, Vol. 28, No. 11, pp. 1657-1672, 1995.

[19] A. Sherstinsky and R. W. Picard, Restoration and Enhancement of Fingerprint Images Using M-Lattice-A Novel Non-Linear Dynamical System, *Proc. 12th ICPR-B*, Jerusalem, pp. 195-200, 1994.

[20] D. B. G. Sherlock, D. M. Monro and K. Millard, Fingerprint Enhancement by Directional Fourier Filtering, *IEE Proc. Vis. Image Signal Processing,* Vol. 141, No. 2, pp. 87-94, 1994.

[21] J. P. P. Starink and E. Backer, Finding Point Correspondence Using Simulated Annealing, *Pattern Recognition,* Vol. 28, No. 2, pp. 231-240, 1995.

[22] G. Stockman, S. Kopstein and S. Benett, Matching Images to Models for Registration and Object Detection via Clustering. *IEEE Transactions on Pattern Analysis and Machine Intelligence,* Vol. 4, No. 3, pp. 229-241, 1982.

[23] J. Ton and A. K. Jain, Registering Landsat Images by Point Matching, *IEEE Transactions on Geoscience and Remote Sensing,* Vol. 27, No. 5, pp. 642-651, 1989.

[24] V. V. Vinod and S. Ghose, Point Matching Using Asymmetric Neural Networks, *Pattern Recognition,* Vol. 26, No. 8, pp. 1207-1214, 1993.

[25] C. I. Watson and C. L. Wilson, NIST Special Database 4, Fingerprint Database, National Institute of Standards and Technology, March 1992.

[26] C. L. Wilson, G. T. Gandela and C. I. Watson, Neural-Network Fingerprint Classification, Journal of Artificial Neural Networks, Vol. 1, No. 2, pp. 203-228, 1994.

[27] Q. Xiao and Z. Bian, An Approach to Fingerprint Identification by Using the Attributes of Feature Lines of Fingerprint, *Proc. 7th ICPR*, Paris, pp. 663-665, 1986.

Core-Based Structure Matching Algorithm of Fingerprint Verification

Weiwei Zhang, Yangsheng Wang
National Laboratory of Pattern Recognition, Institute of Automation,
Chinese Academy of Sciences, Beijing, China, 100080
{ wwzhang, wys }@nlpr.ia.ac.cn

Abstract

Fingerprint matching algorithm is a key issue of the fingerprint recognition, and there already exist many fingerprint matching algorithms, According to the dependence of the core point, fingerprint matching algorithms are divided into two groups, core-based match algorithms and noncore-based match algorithms. Most of the noncore-based matching algorithm is time consuming, therefore, they are not suitable for online application; meanwhile, the core-based matching algorithm is efficient than the noncore-based matching algorithm, but it highly depends on the core detection precision. In this paper, we present a new core-based structure matching algorithm which considers both efficient and precision. Firstly we used core detection algorithm to get the core position, then we define some local structure of the core area. Used these local structure, we can find some correspondent points of the two fingerprint image. Secondly, we use the correspondent points in the first stage to match the global feature of the fingerprint. Experimental results show that the performance of the proposed algorithm is good.

1. Introduction

Fingerprint-based identification has been used for a very long time owning to their uniqueness and immutability, today, fingerprints are the most widely used biometrics features in automatic verification and identification systems. Most Automatic Fingerprint Identification Systems (AFIS) are based on local ridge feature, such as ridge endings and ridge bifurcation [1].

The key issue of the fingerprint recognition is the minutiae matching algorithm. Although there already exist a lot of matching algorithms, most of them are not suitable for the on line application because of following difficulties:

1. The alignment of two minutiae feature vector. After the feature extraction of the fingerprint, we get a minutia feature vector. Before match two feature vectors, we need to align the two feature vectors. According to usage of the core of the fingerprint, the alignment methods can be divided into two groups: core-based alignment and none core-based alignment method.

2. Due to the noise of the fingerprint image, the preprocessing algorithm can not remove all of the noise from the feature vector. Therefore, the match algorithm should robust to the noise.

3. The geometric distortion of the fingerprint image may greatly affect the match algorithm, and there still is no effective method to deal with this nonlinear distortion.

Moreover, the on-line system require the match speed as fast as possible and the template size as small as possible, eg. Just the x y coordinates and orient of the minutiae in some on line system. In order to solve all of the difficulties mentioned above, there are a lot of researches dealing with some of them in literature.

In [12], Z. Chen proposed a topology-based matching algorithm of fingerprint authentication. In their algorithm, they constructed a structure for minutiae in a pre-specified neighborhood, then they used tree matching algorithm to match two fingerprint templates. Their algorithm is invariant to translation, rotation and does not depend on core point, but their algorithm is greatly affected by noise. When the noise increase, the performance of their algorithm degrades rapidly. Further more, they constructed a structure for each minutiae, this will lead to large template. Andrew K Hrechak proposed a structural matching algorithm [11], which used the local structure of the minutia to describe the characteristics of the minutiae, and in [13] some improvements of this approach is proposed. Both of those two methods just used the local structure information of the fingerprint. However the local structure that from same finger may have less similarity due to noise, and they need to build the local structure during the feature extraction stage, which will lead to large size template. Therefore their algorithm is not

1051-4651/02 $17.00 © 2002 IEEE

suitable for the on-line application. In [1], Jain proposed an alignment-based elastic matching algorithm in which the ridge link to the minutiae is used to alignment the feature vector. In order to do this alignment, they save the ten sample points of the ridge which link to the minutiae. This also leads to large size template.

In [5],. Xudong. J Proposed a matching algorithm which based on local and global structure. First they used the k-nearest neighborhood minutiae of each minutia from the local minutia structure, and define a similarity level to measure the similarity of two local structures. Then they used the most similar minutiae pair as the correspondent point to alignment the feature vector, and match the global feature vector considering the local structure similarity. This method has some advantages in processing speed and robustness to rotation. However, because they need to extract the minutiae type and ridge line count between each minutiae and its k-nearest neighborhood exactly, therefore the performance of their algorithm will be greatly affected by the noise. In [6], Nalini K. Ratha proposed a fingerprint authentication using local structural algorithm. First they obtained a minimum set of matched node pairs by matching their neighborhood structures. Then they included more pairs in the match by comparing distances with respect to matched pairs obtained in first phase. Their methods are robustness in large fingerprint database, but their processing speed is low, therefore this method is not suitable for the online application.

When considering an on line application, none of the above mentioned method is suitable for this type application because of the template size and the system efficient. Therefore, in this paper, we propose a new core-based structure matching algorithm which consider both efficient and precision. Firstly we used core detection algorithm to get the core position, and then define some local structure for the minutiae near the core point. Used those local structure, we find some correspondent point of the two fingerprint image. Secondly, we use the correspondent points in the first stage to match the global feature of the fingerprint. The core point detection method is present in section 2. In section 3, the method of match local structure is presented and the global feature matching algorithm is presented in section 4. In section 5, the experiment results of the algorithm are shown. At last, the conclusions are given in section 6.

2. Core point detection

In early works, we have developed a fast multi-resolution based core point detection algorithm [7]. It mainly involves two steps: Firstly, we use the low resolution direction field to localize a singular area which includes core point. Secondly, we use high resolution direction field of singular area to precisely localize the core point. Used this two step method, we can fast and precisely localize the core point in block levels [7].

Due to various type of image quality, our algorithm can not localize the core point in pixel level; therefore, we can not directly use the core point as the reference point for the match algorithm. But for most of the case, the core point detection is very precise, and if we use this reference information, we can greatly reduce the search space, therefore, improve the efficient of the match algorithm. In this paper, we develop a core based matching algorithm, which use the core point to select some minutiae points, and use those minutiae points to construct a lot of local structure. Correspondence point pair can be got by matching those local structures. Use those correspondent points pair, the global minutiae matching can be carried out. The details of the algorithm are given in section 3.

3. Local Structure matching

When an expert recognize a fingerprint, usually he first find correspondences of features and then correlate the features based on minutia type position , orientation and location relative to other features. We usually let the computer do as the human expert does. That is, firstly we used the local structure of the fingerprint to find some correspondence point pairs in local structure matching stage, and then we use the correspondent point pairs to match the global feature vector of the two fingerprints.

As we have known, local structure of the fingerprint minutiae has some properties which help us to match the fingerprint:

1) Local structures in a small area from the same fingerprint are similar in minutiae type, distance and angle.

2) Local structures in a small area that from different fingerprint are often different in minutiae type, distance, angle and topology information.

3) Fingerprints which come from the same finger often have many similar local structures.

4) Fingerprints which come from the different finger often have little similar local structure than those from the same one.

Usually there are 30~60 minutiae in one fingerprint, if we construct local structure for every minutia, the template size will be very large, and large template will decrease the system's efficient. In our algorithm, we only select the minutia which near to the core point of the fingerprint, the selection rule is:

$$|M - Core| < R \qquad (1)$$

Where M and Core denote the minutia and core point separately, | | denote the distance function, R is a constant which is determine by experiment. Usually, 10~15 minutia is enough for the local structure. Therefore, reader can adjust the R according to the minutiae number.

3.1 Definition of Local Structure

In this section, we present the local structure of the minutiae that used in our match algorithm. Usually, there are two ways to construct a local structure:

First, use all of the minutiae around a minutia within a limit distance to make a local structure. The problem of this method is how to decide the radius, as we have known, if the radius is too small, it will include very few minutiae, this will lead false match with the noise influence. And with large radius, the structure will be seriously affected by the elastic distortion. Second, use the k-nearest neighborhood of the minutiae to construct a local structure.

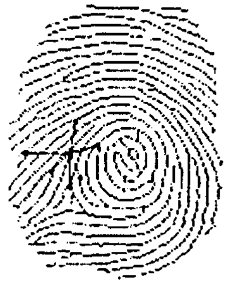
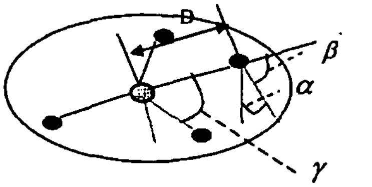


Figure1. A sample of a local structure

In our match algorithm, we used the second method to make a local structure. An example of local structure is shown in figure 1.

Figure 2 shows an example of four-nearest neighbor local structure, where we call the line which connected the center minutiae and the neighbor minutiae as edge. Use the follow information, we construct local structure:

Figure2. Four-nearest neighbor local structure



1) The distance of the center minutiae and the neighbor minutiae, as the D shown in figure 2
2) The angle α between center minutiae's orient and neighbor minutiae's orient in figure 2.
3) The angle β between the minutiae orientation and the edge. As shown in figure 2.
4) The angle γ between the neighboring two edges, as shown in figure 2.

As we have known, the elastic distortion of fingerprint in a local small area often similar, so the distance between the center minutiae and the neighbor minutiae are invariable to translate and rotation. It is apparently that the angle α , β , γ are invariable to translate, rotation. Therefore, we can use those four features to match the local structure.

3.2 The local structure matching algorithm

In this section, we will match the local structure in order to find some correspondent point pairs of the two fingerprints.

There are two methods to match the local structure: The first one is to compute the match cost of the two local structures. Then select the match pair which has the minimal cost as the correspondence pairs.

The second one is to compute the matching edge of the local structure. We can compare the edge from the two local structures. If the edge are similar in distance, angle etc, we say those two edge are match. Moreover, if two edges of one local structure are match with the other two edge of the other local structure, the angles should be similar. Through match all edges of the local structure, we can get the match edge count, and set the match score of the two local structure as the match edge count.

Through experiment we found that the second method are both efficient and robust to noise, therefore we select second as our local structure matching algorithm.

Now we describe our local structure matching algorithm in details. For each edge of the two local structures we compute:

1) The relative distance difference R_d of the two edges, suppose D_1 and D_2 are distance of two edge's from different local structure.

$$R_d = |D_1 - D_2| / \min(D_1, D_2) \qquad (2)$$

The min() function means to select the minimal of two values.

2) The angle difference.

$$D_a = \alpha_1 - \alpha_2$$

$$D_\beta = \beta_1 - \beta_2$$

If R_d , D_a , D_β are below a given threshold, we say the two edges are matching.

3) Two edges of the template which match the other edges of the input image, we compute the difference of the angle γ, denote as D_γ.

$$D_\gamma = \gamma_1 - \gamma_2$$

If the D_γ is below a given threshold, we said that both of those two edges are really match, and cumulate the match score of the two local structures with two.

After those steps, we will get a match score of the two local structures. In order to avoid one edge from a local structure match more than one edge with the other local structure, we set a flag with the compared edge, and just use the edge which hasn't be matched to start the new edge match.

4. Global matching algorithm

In section 3, we match the local structure around the core point, However, just match the local structure is not enough for the verification decision because of the local structure usually tend to similar among different fingerprints. Therefore in this section, we match the global feature based on the local structure match result. From the local structure, we select the two structures which have maximum match score, and use the center minutiae of the two structures as the correspondent point pair of the two fingerprints. Using this correspondent point pair, we normalize the global feature.

Let

$$P = \left((x_1^p, y_1^p, \theta_1^p)^T, \ldots, (x_m^p, y_m^p, \theta_m^p)^T\right)$$

denotes the minutiae feature vector of the template and

$$Q = \left((x_1^q, y_1^q, \theta_1^q)^T, \ldots, (x_n^q, y_n^q, \theta_n^q)^T\right)$$

denotes the minutiae feature vector of the input image. If we denote the reference point as (x^r, y^r, θ^r), we have the normalization formulation:

$$\begin{pmatrix} r_i \\ \theta_i \\ o_i \end{pmatrix} = \begin{pmatrix} \sqrt{(x_i - x^r)^2 + (y_i - y^r)^2} \\ \arctan\left(\dfrac{y_i - y^r}{x_i - x^r}\right) \\ \theta_i - \theta^r \end{pmatrix} \quad (3)$$

Where the (x_i, y_i, θ_i) denote the minutiae from both the template and input image. After the normalization, we match the minutiae vector through an elastic box, as suggest by Jain [1].

In order to avoid one minutia match many minutiae or many minutiae match to one minutia, we construct a match table M for each matched minutiae pair P_i, Q_j and we compute the "distance " of those match minutiae pair, let

$$\Delta d = r_i - r_j$$

$$\Delta \alpha = abs(\theta_i - \theta_j) + abs(o_i - o_j) \quad (4)$$

And save this "distance" into the match table,

$$M(i,j) = \Delta d + \Delta \alpha$$

After the total global match finish, we search the match table, the search rule is:

For each row, if there exists nonzero value, we get the minimum value, and set this column and row to zero except this minimum value. After this operation, the match number is the nonzero count k in the match table. We define a match rate for two fingerprints;

$$R = \sqrt{k * k / m * n} \quad (5)$$

Where R is the match rate, and k is the matched pair count, m, n is the minutiae count from template and input fingerprint image. Through an experience threshold, we can make our decision, that is to say, whether the two fingerprints come from a same finger.

5. Experiment Result and analysis

In order to evaluate our algorithm, we test our algorithm on our fingerprint image database. The image size is 300*300 pixels. Our fingerprint image database includes two parts. Part I is 100 image per finger, 4000 image altogether. Part II is 20 images per finger, 4000 image altogether too. We test the FRR on the Part I, and test FAR on the Part II. Our fingerprint image database includes various type quality fingerprint images, as image shown in figure 3, 4, 5.

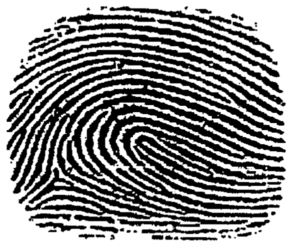


Figure3. Good quality fingerprint image

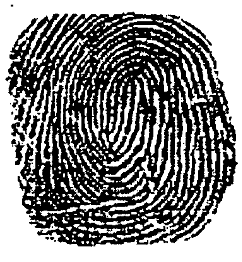


Figure4. Middle quality fingerprint image

We do altogether 80000 times matching. The test results are show at table 1. The comparison with Jain's algorithm [1] is given out in table 3.

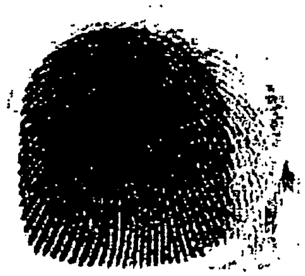


Figure5. Poor quality fingerprint image

Table 1 Fingerprint verification result of our algorithm

	False Accept Rate	False Refuse Rate
1	5.0%	0.08%
2	4.0%	0.09%
3	3.0%	0.35%
4	2.0%	0.58%
5	1.0%	0.80%

Table 2 Fingerprint verification result of Jain' algorithm

	False Accept Rate	False Refuse Rate
1	5.0%	0.8%
2	4.0%	1.7%
3	3.0%	1.8%
4	2.0%	3.2%
5	1.0%	5.6%

From the table we can see that, our algorithm has better performance in our live fingerprint database. Moreover, the average template size is 256 Byte, and the total time consuming of our algorithm is 0.001second on PIII 450M Hz. From the test result we can see that our algorithm has both better performance and efficient, therefore is more suitable for online application.

The reason that leads to the verification failed is the poor quality fingerprint image as shown in figure 5. Those types of fingerprint images are confused even by manual verification. If we add some strict image quality estimate algorithm, the performance can greatly improved, therefore the future work of our system is to develop some strict image quality estimate algorithm as well as improve the minutiae extraction precision.

6. Conclusion

In this paper, we propose a core based structure matching algorithm of fingerprint, which use the minutiae near core point of the fingerprint to construct some local structures. Through matching the local structures, we can get correspondence minutiae. The global match is carried out based on those correspondence minutiae. Experiment results show that the performance and the efficient are significant improved with our algorithm.

7. References

[1] Anil Jain, Lin Hong and Ruud Boole On-Line Fingerprint Verification, IEEE Trans. PAMI, Vol. 19, NO. 4, 1997, 302-314

[2] Nalini K. Ratha, Vinayaka D. Pandit, Robust Fingerprint Authentication Using Local Structure Similarith, IEEE, 2000

[3] Andres Almansa, Laurent Cohen, Fingerprint image matching by minimization of a thin-plate energy using a two-step algorithm with auxiliary vairables, IEEE, 2000

[4] Zsolt Miklos Kovacs-Vajna, A Fingerprint Verification system Based on Triangular Matching and Dynamic Time Warping, IEEE Trans. PAMI, Vol. 22, NO. 11, 2000, pp1266-1276.

[5] A.J.Willis, L..Myers, A Cost-effective fingerprint recognition system for use with low-quality prints and damaged fingerprint. Pattern Recognition , Vol . 34, 2001, pp 255-270

[6] V.s. Srinivasan, N.N. Murthy, Detection of Singular Point in Fingerprint Images, Pattern Recognition, Vol. 25, NO. 2, 1992, pp 139-153.

[7] Weiwei Zhang, Singular Point Detection in fingerprint image, Accept by ACCV 2002

[8] Asker M. Bazen, Sabih H.Gerez, Extraction of Singular Points from Directional Fields of Fingerprints. Annual CTIT Workshop, February 2001, Enschede, Netherlands.

[9] Shih-Hsu Chang, Fang-Hsuan Cheng, Fast Algorithm for Point Pattern Matching: Invariant to Translations, Rotations and Scale Changes, Pattern Recognition, VOL. 30, NO. 2. 1997, pp311-320.

[10] D.K.Isenor, S.G.Zaky, Fingerprint Identification Using Graph matching, Pattern Recognition, Vol. 19, NO. 2, 1986, pp 113-122

[11] Andrew K. Hrechak, James A. Mchugh, Automated Fingerprint Recognition Using Structural Matching, Pattern Recognition, Vol. 23, NO. 8, 1990, pp 839-904.

[12] Z. Chen, CH.Kou, A Toplogy-Based Matching Algorithm for Fingerprint Authentication, IEEE 1991.

[13] A. Wahab, Novel Approach to Automated fingerprint Recongnition, IEE Proceedings, 1998

[14] Kou Chin Fan, Cheng W. Liu and Yuan Kai Wang, A randomized approach with geometric constraints to fingerprint verification, Pattern Recognition, Vol. 33, 2000, 1793-1830.

Extract from:

D. Maltoni, D. Maio, A.K. Jain, S. Prabhakar
Handbook of Fingerprint Recognition
Springer, New York, 2003

3.6: Singularity and Core Detection
(extract)

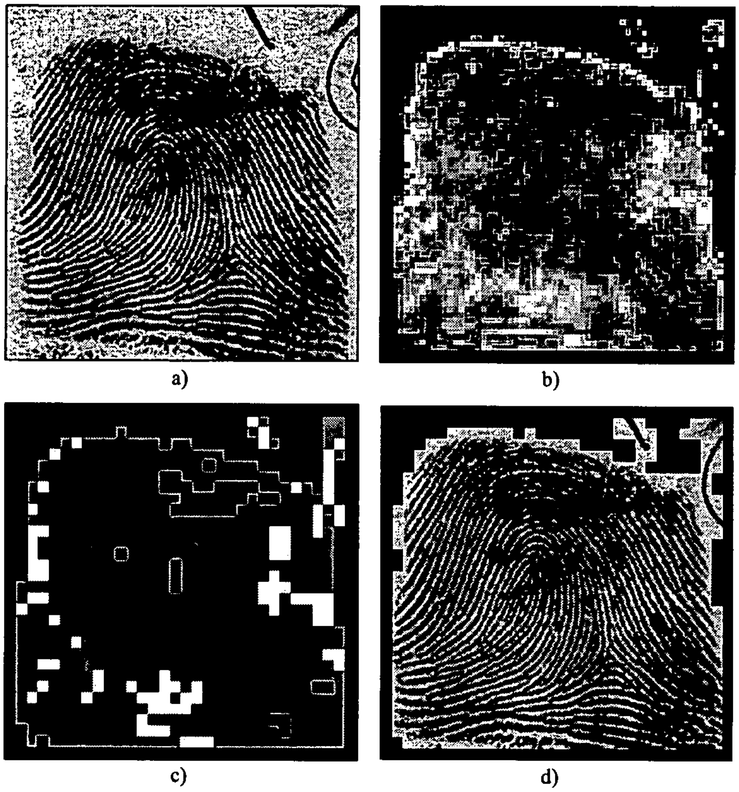


Figure 3.13. Segmentation of a fingerprint image as proposed by Ratha, Chen, and Jain (1995): a) original image; b) variance field; c) quality image derived from the variance field: a quality value "good," "medium," "poor" or "background" is assigned to each block according to its variance; d) segmented image. ©Elsevier.

3.6 Singularity and Core Detection

Most of the approaches proposed in the literature for singularity detection operate on the fingerprint orientation image. In the rest of this section, the main approaches are coarsely classified and a subsection is dedicated to each family of algorithms.

Poincaré method index

An elegant and practical method based on the Poincaré index was proposed by Kawagoe and Tojo (1984). Let **G** be a vector field and *C* be a curve immersed in **G**; then the Poincaré index $P_{G,C}$ is defined as the total rotation of the vectors of **G** along *C* (see Figure 3.14).

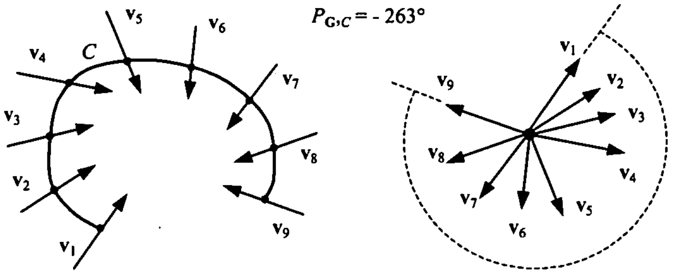


Figure 3.14. The Poincaré index computed over a curve *C* immersed in a vector field **G**.

Let **G** be the field associated with a fingerprint orientation image[1] **D** and let $[i,j]$ be the position of the element θ_{ij} in the orientation image; then the Poincaré index $P_{G,C}(i,j)$ at $[i,j]$ is computed as follows.

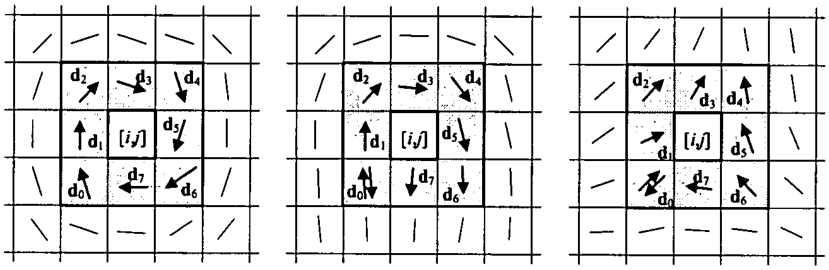
- The curve *C* is a closed path defined as an ordered sequence of some elements of **D**, such that $[i,j]$ is an internal point;
- $P_{G,C}(i,j)$ is computed by algebraically summing the orientation differences between adjacent elements of *C*. Summing orientation differences requires a direction (among the two possible) to be associated at each orientation. A solution to this problem is to randomly select the direction of the first element and assign the direction closest to that of the previous element to each successive element. It is well known and can be easily shown that, on closed curves, the Poincaré index assumes only one of the discrete values: 0°, ±180°, and ±360°. In the case of fingerprint singularities:

$$P_{G,C}(i,j)=\begin{cases} 0° & \text{if } [i,j] \text{ does not belong to any singular region} \\ 360° & \text{if } [i,j] \text{ belongs to a whorl type singular region} \\ 180° & \text{if } [i,j] \text{ belongs to a loop type singular region} \\ -180° & \text{if } [i,j] \text{ belongs to a delta type singular region.} \end{cases}$$

[1] Note that a fingerprint orientation image is not a true vector field inasmuch as its elements are unoriented directions.

Figure 3.15 shows three portions of orientation images. The path defining C is the ordered sequence of the eight elements $\mathbf{d}_k$ (k = 0..7) surrounding $[i,j]$. The direction of the elements $\mathbf{d}_k$ is chosen as follows: $\mathbf{d}_0$ is directed upward; $\mathbf{d}_k$ (k = 1..7) is directed so that the absolute value of the angle between $\mathbf{d}_k$ and $\mathbf{d}_{k-1}$ is less than or equal to 90°. The Poincaré index is then computed as

$$P_{G,C}(i,j) = \sum_{k=0..7} angle\left(\mathbf{d}_k, \mathbf{d}_{(k+1)\bmod 8}\right).$$



$P_{G.C}(i,j) = 360°$ $P_{G.C}(i,j) = 180°$ $P_{G.C}(i,j) = -180°$

Figure 3.15. Example of computation of the Poincaré index in the 8-neighborhood of points belonging (from the left to the right) to a whorl, loop, and delta singularity, respectively. Note that for the loop and delta examples (center and right), the direction of $\mathbf{d}_0$ is first chosen upward (to compute the angle between $\mathbf{d}_0$ and $\mathbf{d}_1$) and then successively downward (when computing the angle between $\mathbf{d}_7$ and $\mathbf{d}_0$).

An example of singularities detected by the above method is shown in Figure 3.16.a.

An interesting implementation of the Poincaré method for locating singular points was proposed by Bazen and Gerez (2002b): according to Green's theorem, a closed line integral over a vector field can be calculated as a surface integral over the rotation of this vector field; in practice, instead of summing angle differences along a closed path, the authors compute the "rotation" of the orientation image (through a further differentiation) and then perform a local integration (sum) in a small neighborhood of each element. Bazen and Gerez (2002b) also provided a method for associating an orientation with each singularity; this is done by comparing the orientation image around each detected singular point with the orientation image of an ideal singularity of the same type.

Singularity detection in noisy or low-quality fingerprints is difficult and the Poincaré method may lead to the detection of false singularities (Figure 3.17). Regularizing the orientation image through a local averaging, as discussed in Section 3.3, is often quite effective in preventing the detection of false singularities.

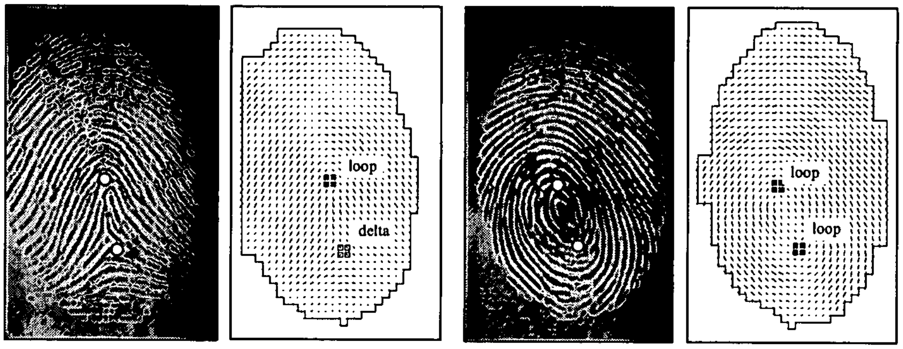


Figure 3.16. Singularity detection by using the Poincaré index method. The elements whose Poincaré index is 180° (loop) or -180° (delta) are enclosed by small boxes. Usually, more than one point (four points in these examples) is found for each singular region: hence, the center of each singular region can be defined as the barycenter of the corresponding points.

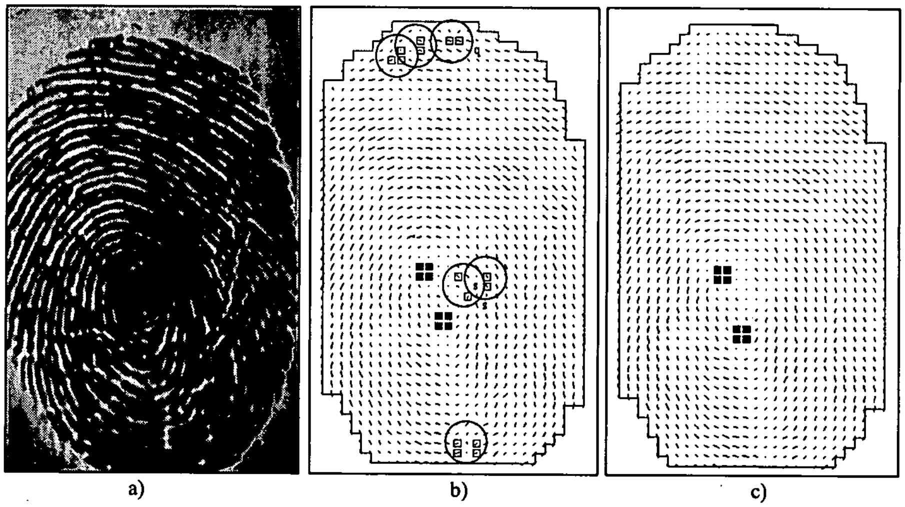


Figure 3.17. a) A poor quality fingerprint; b) the singularities of the fingerprint in a) are extracted through the Poincaré method (circles highlight the false singularities); c) the orientation image has been regularized and the Poincaré method no longer provides false alarms.

A Survey of Image Registration Techniques

LISA GOTTESFELD BROWN

Department of Computer Science, Columbia University, New York, NY 10027

Registration is a fundamental task in image processing used to match two or more pictures taken, for example, at different times, from different sensors, or from different viewpoints. Virtually all large systems which evaluate images require the registration of images, or a closely related operation, as an intermediate step. Specific examples of systems where image registration is a significant component include matching a target with a real-time image of a scene for target recognition, monitoring global land usage using satellite images, matching stereo images to recover shape for autonomous navigation, and aligning images from different medical modalities for diagnosis.

Over the years, a broad range of techniques has been developed for various types of data and problems. These techniques have been independently studied for several different applications, resulting in a large body of research. This paper organizes this material by establishing the relationship between the variations in the images and the type of registration techniques which can most appropriately be applied. Three major types of variations are distinguished. The first type are the variations due to the differences in acquisition which cause the images to be misaligned. To register images, a spatial transformation is found which will remove these variations. The class of transformations which must be searched to find the optimal transformation is determined by knowledge about the variations of this type. The transformation class in turn influences the general technique that should be taken. The second type of variations are those which are also due to differences in acquisition, but cannot be modeled easily such as lighting and atmospheric conditions. This type usually effects intensity values, but they may also be spatial, such as perspective distortions. The third type of variations are differences in the images that are of interest such as object movements, growths, or other scene changes. Variations of the second and third type are not directly removed by registration, but they make registration more difficult since an exact match is no longer possible. In particular, it is critical that variations of the third type are not removed. Knowledge about the characteristics of each type of variation effect the choice of feature space, similarity measure, search space, and search strategy which will make up the final technique. All registration techniques can be viewed as different combinations of these choices. This framework is useful for understanding the merits and relationships between the wide variety of existing techniques and for assisting in the selection of the most suitable technique for a specific problem.

Categories and Subject Descriptors: A.1 [**General Literature**]: Introductory and Survey; I.2.10 [**Artificial Intelligence**]: Vision and Scene Understanding; I.4 [**Computing Methodologies**]: Image Processing; I.5 [**Computing Methodologies**]: Pattern Recognition

General Terms: Algorithms, Design, Measurement, Performance

Additional Key Words and Phrases: Image registration, image warping, rectification, template matching

CONTENTS

1. INTRODUCTION

A frequent problem arises when images taken, at different times, by different sensors or from different viewpoints need to be compared. The images need to be aligned with one another so that differences can be detected. A similar problem occurs when searching for a prototype or template in another image. To find the optimal match for the template in the image, the proper alignment between the image and template must be found. All of these problems, and many related variations. are solved by methods that perform image registration. A transformation must be found so that the points in one image can be related to their corresponding points in the other. The determination of the optimal transformation for registration depends on the types of variations between the images. The objective of this paper is to provide a framework for solving image registration tasks and to survey the classical approaches.

Registration methods can be viewed as different combinations of choices for the following four components:

(1) a feature space,

(2) a search space,

(3) a search strategy, and

(4) a similarity metric.

The *feature space* extracts the information in the images that will be used for matching. The *search space* is the class of transformations that is capable of aligning the images. The *search strategy* decides how to choose the next transformation from this space, to be tested in the search for the optimal transformation. The *similarity metric* determines the relative merit for each test. Search continues according to the search strategy until a transformation is found whose similarity measure is satisfactory. As we shall see, the types of variations present in the images will determine the selection for each of these components.

For example, consider the problem of registering the two x-ray images of chest taken of the same patient at different times shown in Figure 1. Properly aligning the two images is useful for detecting, locating, and measuring pathological and other physical changes. A standard approach to registration for these images might be as follows: the images might first be reduced to binary images by detecting the edges or regions of highest contrast using a standard edge detection scheme. This removes extraneous information and reduces the amount of data to be evaluated. If it is thought that the primary difference in acquisition of the images was a small translation of the scanner, the search space might be a set of small translations. For each translation of the edges of the left image onto the edges of the right image, a measure of similarity would be computed. A typical similarity measure would be the correlation between the images. If the similarity measure is computed for all translations then the search strategy is simply exhaustive. The images are registered using the translation which optimizes the similarity criterion. However, the choice of using edges for features, translations for the search space, exhaustive search for the search strategy and correlation for the similarity metric will influence the outcome of this registration. In fact, in this case, the registration will undoubtably be unsatisfactory since the images are misaligned in a more complex

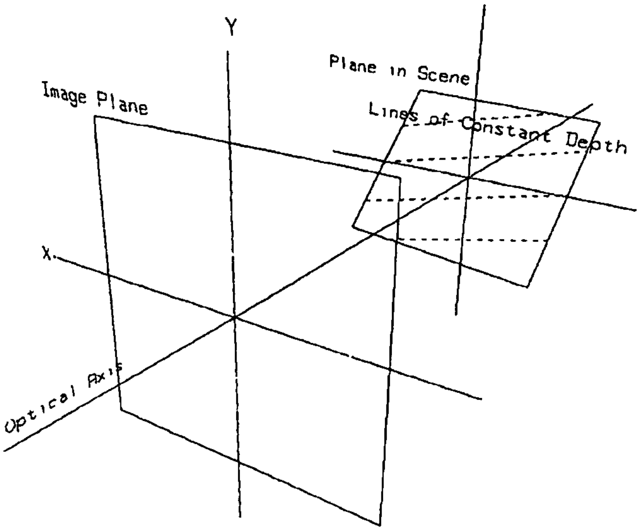


Figure 6. Any plane can be decomposed into lines parallel to the image plane.

where A, B, and C are constants, rectification can be performed by mapping the intensity of the image point at (x_i, y_i) into the new rectified image point location $(fx_i/Z, fy_i/Z)$ where $Z = f - Ax_i - By_i$ [Rosenfeld 1982]. This is because the scene plane can be decomposed into lines $Ax_o + By_o = C'$ each at a constant distance $(z_o = C - C')$ from the image plane. Each line then maps to a line in the image plane, and since its perspective distortion is related to its distance from the image, all points on this line must be scaled accordingly by $f/(C - C' - f)$. Figure 6 shows how a plane is decomposed into lines that are each parallel to the image plane.

Two pictures of the flat plane taken from different viewpoints can be registered by the following steps. First, the scene coordinates (x_1, y_1, z_1) are related to their image coordinates in image 1 of a point with respect to camera 1 by a scale factor $(z_1 - f)/f$ dependent on their depth (the z_1 coordinate) and the lens center f because of similar triangles. This gives us two equations. Since they must also satisfy the equation of the plane, we have three equations from which we can derive the three coordinates of each scene point using its corresponding image point with respect to coordinate system of camera 1. The scene coordinates are then converted from the coordinate system with respect to camera 1 to a coordinate system with respect to camera 2 to obtain (x_2, y_2, z_2). Lastly, these can be projected onto image 2 by the factor $f/(z_2 - f)$, again by similar triangles. Of course, if these are discrete images, there is still the problem of interpolation if the registered points do not fall on grid locations. See Wolberg [1990] for a good survey of interpolation methods.

3. REGISTRATION METHODS

3.1 Correlation and Sequential Methods

Cross-correlation is the basic statistical approach to registration. If is often used for template matching or pattern recognition in which the location and orientation of a template or pattern is found in a picture. By itself, cross-correlation is not a registration method. It is a similarity measure or match metric, i.e., it gives a measure of the degree of similarity between an image and a template. However, there are several registration methods for which it is the primary tool, and it is these methods and the closely related sequential methods which are discussed in this section. These methods are generally useful for images which are misaligned by small rigid or affine transformations.

For a template T and image I, where T is small compared to I, the two-dimensional normalized cross-correlation function measures the similarity for each translation:

$$C(u,v) = \frac{\Sigma_x \Sigma_y T(x,y) I(x-u, y-v)}{\sqrt{\left[\Sigma_x \Sigma_y I^2(x-u, y-v)\right]}}.$$

If the template matches the image exactly, except for an intensity scale factor, at a translation of (i,j), the cross-correlation will have its peak at $C(i,j)$. (See Rosenfeld and Kak [1982] for a proof of this using the Cauchy-Schwarz inequality.) Thus, by computing C over all possible translations, it is possible to find the degree of similarity for any template-sized window in the image. Notice the cross-correlation must be normalized since local image intensity would otherwise influence the measure.

The cross-correlation measure is directly related to the more intuitive measure which computes the sum of the differences squared between the template and the picture at each location of the template:

$$D(u,v) = \sum_x \sum_y (T(x,y) - I(x-u, y-v))^2.$$

This measure decreases with the degree of similarity since, when the template is placed over the picture at the location (u,v) for which the template is most similar, the differences between the corresponding intensities will be smallest. The template energy defined as $\Sigma_x \Sigma_y T^2(x,y)$ is constant for each position (u,v) that we measure. Therefore, we should normalize, as before, using the local image energy $\Sigma_x \Sigma_y I^2(x-u, y-v)$. Notice that if you expand this intuitive measure $D(u,v)$ into its quadratic terms, there are three terms: a template energy term, a product term of template and image, and an image energy term. It is the product term or correlation $\Sigma_x \Sigma_y T(x,y) I(x-u, y-v)$ which when normalized, determines the outcome of this measure.

A related measure, which is advantageous when an absolute measure is needed, is the correlation coefficient

$$\frac{covariance(I,T)}{\sigma_I \sigma_t} = \frac{\Sigma_x \Sigma_y (T(x,y) - \mu_T)(I(x-u, y-v) - \mu_I)}{\sqrt{\Sigma_x \Sigma_y (I(x-u, y-v) - \mu_I)^2 \Sigma_x \Sigma_y (T(x,y) - \mu_T)^2}}$$

where μ_T and σ_T are mean and standard deviation of the template and μ_I and σ_I are mean and standard deviation of the image.[1] This statistical measure has the property that it measures correlation on an absolute scale ranging from $[-1, 1]$. Under certain statistical assumptions, the value measured by the correlation coefficient gives a linear indication of the similarity between images. This is useful in order to quantitatively measure confidence or reliability in a match and to reduce the number of measurements needed when a prespecified confidence is sufficient [Svedlow et al. 1976].

Consider a simple example of a binary image and binary template, i.e., all the pixels are either black or white, for which it is possible to predict with some probability whether or not a pixel in the image will have the same binary value as a pixel in the template. Using the correlation coefficient, it is possible to compute the probability or confidence that the image is an instance of the template. We assume the template is an ideal repre-

[1] The mean μ of an image is the average intensity value; if the image I is defined over a region $x = 1, N$; $y = 1, M$ then $\mu_I = \Sigma_{x=1}^{N} \Sigma_{y=1}^{M} (I(x, y)/(N * M))$. The standard deviation is a measure of the variation there in the intensity values. It is defined as $\sigma_I^2 = \Sigma_{x=1}^{N} \Sigma_{y=1}^{M} ((I(x, y) - \mu_I)^2/(N * M))$.

sentation of the pattern we are looking for. The image may or may not be an instance of this pattern. However, if we can statistically characterize the noise that has corrupted the image, then the correlation coefficient can be used to quantitatively measure how likely it is that the image is an instance of the template.

Another useful property of correlation is given by the Correlation theorem. The Correlation theorem states that the Fourier transform of the correlation of two images is the product of the Fourier transform of one image and the complex conjugate of the Fourier transform of the other. This theorem gives an alternate way to compute the correlation between images. The Fourier transform is simply another way to represent the image function. Instead of representing the image in the spatial domain, as we normally do, the Fourier transform represents the same information in the frequency domain. Given the information in one domain we can easily convert to the other domain. The Fourier transform is widely used in many disciplines, both in cases where it is of intrinsic interest and as a tool, as in this case. It can be computed efficiently for images using the Fast Fourier Transform or FFT. Hence, an important reason why the correlation metric is chosen in many registration problems is because the Correlation theorem enables it to be computed efficiently, with existing, well-tested programs using the FFT (and occasionally in hardware using specialized optics). The use of the FFT becomes most beneficial for cases where the image and template to be tested are large. However there are two major caveats. Only the cross-correlation before normalization may be treated by FFT. Second, although the FFT is faster it also requires a memory capacity that grows with the log of the image area. Last, both direct correlation and correlation using FFT have costs which grow at least linearly with the image area.

Solving registration problems like template matching using correlation has many variations [Pratt 1978]. Typically the cross-correlation between the image and the template (or one of the related similarity measures given above) is computed for each allowable transformation of the template. The transformation whose cross-correlation is the largest specifies how the template can be optimally registered to the image. This is the standard approach when the allowable transformations include a small range of translations, rotations, and scale changes; the template is translated, rotated, and scaled for each possible translation, rotation, and scale of interest. As the number of transformations grows, however, the computational costs quickly become unmanageable. This is the reason that the correlation methods are generally limited to registration problems in which the images are misaligned only by a small rigid or affine transformation. In addition, to reduce the cost of each measurement for each transformation instance, measures are often computed on features instead of the whole image area. Small local features of the template which are more invariant to shape and scale, such as edges joined in a Y or a T, are frequently used.

If the image is noisy, i.e., there are significant distortions which cannot be removed by the transformation, the peak of the correlation may not be clearly discernible. The Matched Filter Theorem states that for certain types of noise such as additive white noise, the cross-correlation filter that maximizes the ratio of signal power to the expected noise power of the image, i.e., the information content, is the template itself. In other cases however, the image must be prefiltered before cross-correlation to maintain this property. The prefilter and the cross-correlation filter (the template) can be used to produce a single filter which can simultaneously perform both filtering operations. The prefilter to be used can sometimes be determined if the noise in the image satisfies certain statistical properties. These techniques, which prefilter based on the properties of the noise of the image in order to maximize the peak correlation with respect to this noise (using the Matched Filter Theorem) and then cross-correlate, are called *matched filter* techniques [Rosenfeld and Kak 1982]. The disadvantages of these techniques are that they can be computationally intensive, and in practice the statistical assumptions about the noise in the image are difficult to satisfy.

A far more efficient class of algorithms than traditional cross-correlation, called the sequential similarity detection algorithms (SSDAs), was proposed by Barnea and Silverman [1972]. Two major improvements are offered. First, they suggest a similarity measure $E(u,v)$, which is computationally much simpler, based on the absolute differences between the pixels in the two images,

$$E(u,v) = \sum_x \sum_y |T(x,y) - I(x-u, y-v)|.$$

The normalized measure is defined as

$$E(u,v) = \sum_x \sum_y |T(x,y) - \hat{T} - I(x-u, y-v) + \hat{I}(u,v)|$$

where $\hat{T}$ and $\hat{I}$ are the average intensities of the template and local image window respectively. This is significantly more efficient than correlation. Correlation requires both normalization and the added expense of multiplications. Even if this measure is unnormalized a minimum is guaranteed for a perfect match. Normalization is useful, however, to get an absolute measure of how the two images differ, regardless of their intensity scales.

The second improvement Barnea and Silverman [1972] introduce is a sequential search strategy. In the simplest case of translation registration this strategy might be a sequential thresholding. For each window of the image (determined by the translation to be tested and the template size), one of the similarity measures defined above is accumulated until the threshold is exceeded. For each window the number of points that were examined before the threshold was exceeded is recorded. The window which examined the most points is assumed to have the lowest measure and is therefore the best registration.

The sequential technique can significantly reduce the computational complexity with minimal performance degradation. There are also many variations that can be implemented in order to adapt the method to a particular set of images to be registered. For example, an ordering algorithm can be used to order the windows tested which may depend on intermediate results, such as a coarse-to-fine search or a gradient technique. These strategies will be discussed in more detail in Section 4.3. The ordering of the points examined during each test can also vary depending on critical features to be tested in the template. The similarity measure and the sequential decision algorithm might vary depending on the required accuracy, acceptable speed, and complexity of the data.

Although the sequential methods improve the efficiency of the similarity measure and search, they still have increasing complexity as the degrees of freedom of the transformation is increased. As the transformation becomes more general the size of the search grows. On the one hand, sequential search becomes more important in order to maintain reasonable time complexity; on the other hand it becomes more difficult not to miss good matches.

In comparison with correlation, the sequential similarity technique improves efficiency by orders of magnitude. Tests conducted by Barnea and Silverman [1972], however, also showed differences in results. In satellite imagery taken under bad weather conditions, clouds needed to be detected and replaced with random noise before correlation would yield a meaningful peak. Whether the differences found in their small study can be extended to more general cases remains to be investigated.

A limitation of both of these methods is their inability to deal with dissimilar images. The similarity measures described so far, the correlation coefficient, and the sum of absolute differences are maximized and minimized, respectively for identical matches. For this reason, feature-based techniques and measures based on the invariant properties of the Fourier transform are preferable when images are acquired under different circumstances, e.g., varying lighting or atmospheric conditions. In the next section the Fourier methods will be described. Like the correlation and sequential methods, the Fourier methods are appropriate for small translations, rotations, or scale changes. The correlation methods can be used sometimes for more general rigid transformations but become inefficient as the degrees of freedom of the transformation grows. The Fourier methods can only be used where the Fourier transform of an image which has undergone the transformation is related in a nice mathematical way to the original image. The methods to be described in the next section are applicable for images which have been translated or rotated or both. They are specifically well suited for images with low frequency or frequency-dependent noise; lighting and atmospheric variations often cause low-frequency distortions. They are not appropriate for images with frequency-independent noise (white noise) or for more general transformations.

3.2 Fourier Methods

The methods to be described in this section register images by exploiting several nice properties of the Fourier Transform. Translation, rotation, reflection, distributivity, and scale all have their counterpart in the Fourier domain. Furthermore, as mentioned in the previous section, the transform can be efficiently implemented in either hardware or using the Fast Fourier Transform. These methods differ from the methods in the last section because they search for the optimal match according to information *in the frequency domain*. The method described in Section 3.1 used the Fourier Transform as a tool to perform a spatial operation, namely correlation.

By using the frequency domain, the Fourier methods achieve excellent robustness against correlated and frequency-dependent noise. They are applicable, however, only for images which have been at most rigidly misaligned. In this section we will first describe the most basic method which uses Fourier Analysis. It is called phase correlation and can be used to register images which have been shifted relative to each other. Then we will describe an extension to this method and several related methods which handle images which have been both shifted and rotated with respect to each other.

Kuglin and Hines [1975] proposed an elegant method, called *phase correlation*, to align two images which are shifted relative to one another. In order to describe their method, we will define a few of the terms used in Fourier Analysis which we will need. The Fourier transform of an image $f(x, y)$ is a complex function; each function value has a real part $R(\omega_x, \omega_y)$ and an imaginary part $I(\omega_x, \omega_y)$ at each frequency (ω_x, ω_y) of the frequency spectrum:

$$F(\omega_x, \omega_y) = R(\omega_x, \omega_y) + iI(\omega_x, \omega_y)$$

where $i = \sqrt{-1}$. This can be expressed alternatively using the exponential form as

$$F(\omega_x, \omega_y) = |F(\omega_x, \omega_y)| e^{i\phi(\omega_x, \omega_y)}$$

where $|F(\omega_x, \omega_y)|$ is the magnitude or amplitude of the Fourier transform and where $\phi(\omega_x, \omega_y)$ is the phase angle. The square of the magnitude is equal to the amount of energy or power at each frequency of the image and is defined as:

$$|F(\omega_x, \omega_y)|^2 = R^2(\omega_x, \omega_y) + I^2(\omega_x, \omega_y).$$

The phase angle describes the amount of phase shift at each frequency and is defined as

$$\phi(\omega_x, \omega_y) = \tan^{-1}\left[I(\omega_x, \omega_y)/R(\omega_x, \omega_y)\right].$$

Phase correlation relies on the translation property of the Fourier transform,